CLOUD COMPUTING 2010

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

November 21-26, 2010 - Lisbon, Portugal

ComputationWorld 2010 Editors

Ali Beklen, IBM Turkey, Turkey
Jorge Ejarque, Barcelona Supercomputing Center, Spain
Wolfgang Gentzsch, EU Project DEISA, Board of Directors of OGF, Germany
Teemu Kanstren, VTT, Finland
Arne Koschel, Fachhochschule Hannover, Germany
Yong Woo Lee, University of Seoul, Korea
Li Li, Avaya Labs Research - Basking Ridge, USA
Michal Zemlicka, Charles University - Prague, Czech Republic

CLOUD COMPUTING 2010

Foreword

The First International Conference on Cloud Computing, GRIDs, and Virtualization [CLOUD COMPUTING 2010], held between November 21 and 26 in Lisbon, Portugal, inaugurated an event under the umbrella of ComputationWorld 2010 intended 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.

We take here the opportunity to warmly thank all the members of the CLOUD COMPUTING 2010 Technical Program Committee, as well as the numerous reviewers. The creation of such a broad and high quality conference program would not have been possible without their involvement. We also kindly thank all the authors who dedicated much of their time and efforts to contribute to CLOUD COMPUTING 2010. We truly believe that, thanks to all these efforts, the final conference program consisted of top quality contributions.

Also, this event could not have been a reality without the support of many individuals, organizations, and sponsors. We are grateful to the members of the CLOUD COMPUTING 2010 organizing committee for their help in handling the logistics and for their work to make this professional meeting a success.

We hope that CLOUD COMPUTING 2010 was a successful international forum for the exchange of ideas and results between academia and industry and for the promotion of progress in the areas of cloud computing, GRIDs, and virtualization.

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

CLOUD COMPUTING 2010 Chairs:

David Bernstein, Huawei, USA
Jorge Ejarque, Barcelona Supercomputing Center, Spain
Wolfgang Gentzsch, EU Project DEISA, Board of Directors of OGF, Germany
Daniel S. Katz, University of Chicago & Argonne National Laboratory, USA
Dieter Kranzlmueller, LMU & LRZ - Munich, Germany
Yong Woo Lee, University of Seoul, Korea
Geng Lin, Cisco Systems, Inc., USA
Leslie Liu, IBM T.J Watson Research, USA
Tiziana Margaria, University of Potsdam, Germany
Tony Shan, Keane Inc., USA
Aameek Singh, IBM Research - Almaden, USA
Kerry Taylor, CSIRO ICT Centre, Australia
Wolf Zimmermann, University of Halle, Germany

CLOUD COMPUTING 2010

Committee

CLOUD COMPUTING Advisory Chairs

Academia
Tiziana Margaria, University of Potsdam, Germany
Daniel S. Katz, University of Chicago & Argonne National Laboratory, USA
Yong Woo Lee, University of Seoul, Korea
Kerry Taylor, CSIRO ICT Centre, Australia
Wolf Zimmermann, University of Halle, Germany

Industry
Geng Lin, Cisco Systems, Inc., USA
Wolfgang Gentzsch, EU Project DEISA, Board of Directors of OGF, Germany
Tony Shan, Keane Inc., USA
David Bernstein, Huawei, USA

Research Institutes
Jorge Ejarque, Barcelona Supercomputing Center, Spain
Dieter Kranzlmueller, LMU & LRZ - Munich, Germany
Aameek Singh, IBM Research - Almaden, USA
Leslie Liu, IBM T.J Watson Research, USA

CLOUD COMPUTING 2010 Technical Program Committee

Stefan Andrei, Lamar University, USA
Janaka Balasooriya, Arizona State University, USA
Ali Beklen, IBM Turkey - Software Group, Turkey
Simona Bernardi, Università di Torino, Italia
Athman Bouguettaya, CSIRO, Australia
Mario Bravetti, Università di Bologna - Cesena, Italy
Jian-Nong Cao, Hong Kong Polytechnic University, Hong Kong
Juan-Vicente Capella-Hernández, Universidad Politécnica de Valencia, Spain
Ying Chen, IBM Research - Beijing, China
William Cheng-Chung Chu, Tunghai University, Taiwan
Dickson Chiu, Dickson Computer Systems, Hong Kong
You Yuan Chwen, Creative Entrepreneurship Consulting, Inc., Taiwan
Bruno Ciciani, University of Rome "La Sapienza", Italy
Ernesto Damiani, Università degli Studi di Milano, Italy
Wael William Diab, Broadcom, USA
Javier Diaz, Universidad de Castilla-La Mancha - Ciudad Real, Spain
Tharam Dillon, Curtin Univerisity of Technology, Australia
Jorge Ejarque, Barcelona Supercomputing Center, Spain

Norbert Ritter, University of Hamburg, Germany
Ivan Rodero, Rutgers the State University of New Jersey / NSF Center for Autonomic Computing, USA
Josef Schiefer, UC4 SENACTIVE Software Inc., Austria
Tony Shan, Keane Inc., USA
Aameek Singh, IBM Research - Almaden, USA
Ben Kwang-Mong Sim, Gwangju Institute of Science & Technology, South Korea
Vladimir Stantchev, Berlin Institute of Technology, Germany / University of California - Berkeley, USA
Wilfried Steiner, TTTech Computertechnik AG, Austria
Azzelarabe Taleb-Bendiab, Liverpool John Moores University, UK
Mallik Tatipamula, Juniper Networks, USA
Kerry Taylor, CSIRO ICT Centre, Australia
Michael Tighe, University of Western Ontario, Canada
George K. Thiruvathukal, Loyola University Chicago, USA
Naohiko Uramoto, IBM Research - Tokyo, Japan
Robert van Engelen, Florida State University, USA
Cho-Li Wang, The University of Hong Kong, China
Fei-Yue Wang, Chinese Academy of Sciences, China
Andy Ju An Wang, Southern Polytechnic State University, USA
Hongbing Wang, Southeast University, China
Yong Woo Lee, University of Seoul, Korea
Xinyu Xu, Sharp Labs of America, USA
Hongji Yang, De Montfort University - Leicester, UK
Qi Yu, Rochester Institute of Technology, USA
Michael Zapf, University of Kassel, Germany
Gianluigi Zavattaro, Università di Bologna, Italy
Weimin Zheng, Tsinghua University - Beijing, China
Zibin Zheng, The Chinese University of Hong Kong, Hong Kong, China
Haibin Zhu, Nipissing University, Canada
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

GUISET LogOn: Design and Implementation of GUISET-driven Authorization Framework.

Obeten O. Ekabua
Dept. of Computer Science and Info. Systems,
University of Venda, Private Bag X5050,
Thohoyandou 0950, South Africa
Email:obeten.ekabua@univen.ac.za

Matthew O. Adigun
Department of Computer Science,
University of Zululand, Private Bag X1001,
KwaDlangezwa 3886, South Africa
Email: matthewo@pan.unizul.ac.za

***Abstract* - Authorization is an important part of GRID security systems with each GRID domain having its own policies that may change dynamically. Authorization ensures that resources can be accessed only by parties who have the appropriate privileges. Many authorization frameworks exist, but these are not applicable to our GUISET (GRID-based Utility Infrastructure for SMME enabled Enterprise Technology) domain. Therefore in this research as reported, we have developed and implemented a GUISET-driven framework, as a security gatekeeper, that satisfies access and privacy requirements for service requesters and providers in GUISET environment.**

Keywords - Authorization; Authentication; GUISET; GRID; Security; Service.

I. INTRODUCTION

As the days goes by, Web Services are being echoed as a solution to the next generation enterprise integration. A large scale service-oriented computing environment like GRID consists of many computers, storage systems and other devices distributed over heterogeneous wide area networks, but presents unique security problems that are not addressed by traditional client-server/distributed computing environments. While providing basic security requirements like authentication, authorization, confidentiality and integrity, the security infrastructure for GRID and Web Services must also be able to support more advanced security features like dynamic delegation of access rights [1].

Using personal sensitive information so as to gain access to a resource in a distributed environment raises an interesting paradox. Firstly, in order to make the services and resources accessible to legitimate users, the authorization infrastructure requires the users' attributes. Secondly, the users may not be ready to disclose their attributes to a remote service provider without determining exactly who the provider is and how personal attributes will be used [2].

GRID refers to the collection of distributed (networked) computers that are geographically dispersed, pooling resources together in such a way that users may utilize processing, storage, software and data resource from any of interconnected computers, leading to greater resource sharing and higher utilization ratio. Therefore, GRID may be viewed as virtual organisations (VO) [3]. Thus, a GRID system is a VO comprising several independent autonomous domains. The security of the GRID system should provide the same protection that conventional systems provide, including establishing the identity of users or services (authentication), protecting communications (encryption/decryption), determining who is allowed to perform what actions (authorization), and recording the important operations processed by the systems (auditing) [4] . GRID can give VO members direct access to each other's computers, programs, files,

data, and networks. Therefore, the sharing of resource in VO must be controlled, secure, flexible, and usually time-limited. The need to support the integration and management
of resources within VO introduces challenging security issues in GRID environment [4].

Authorization is an important part of GRID system, in which every domain may have its own policy and may change its policy dynamically. Hence, the authorization mechanism of GRID computing platform needs to support multiple security policies and need to have flexibility to support dynamism in security policies [4, 5].

Security is becoming increasingly significant when integrating services across multiple VOs. The different resources in a GRID have different access policies, including how they authorize users in accessing those resources or services [5]. Many authorization mechanisms exist including role-based authorization, rule-based authorization, and identity-based authorization. But these authorization mechanisms alone cannot satisfy the access requirements of distributed services as the access depends on many other factors like privacy requirements of the requester, authentication requirements of the service, trust relationship with the requester, authorization and management policies among participating parties, etc. [1].

Authorization ensures that resources can be accessed only by parties who have the appropriate privileges. This makes the resource gatekeeper to require that some level of trust be established before sensitive information can be released. Service requesters are required to submit sufficient authorization credentials before access will be granted. Wherever people are involved in the exchange of digital information, such as credentials, privacy becomes an issue of some concern [2]. Authorization in a distributed environment should be determined as a result of evaluating the request of an authenticated user against various policies like privacy policy, trust policy, authorization policy. Many authorization mechanisms for large scale distributed systems like GRID and Web

 ISBN: 978-1-61208-106-9

ignore one of the components from privacy, trust and policy [1].

The significance of small, micro and medium-sized enterprises SMME in stimulating economic growth, generating economic growth, generating employment, creating social cohesion, as well as regional and local development in Africa is almost undisputed. Throughout the continent SMME promotion is priority in the policy agenda of most African countries as its contribution to poverty alleviation and economic development is globally recognized [3].

The GRID-based Utility Infrastructure for SMME enabled Technology (GUISET) is an infrastructure used to solve barriers experienced by SMME in Africa due to the un-affordability of the technology to run them. The concept of GUISET is based on the idea that there is indeed a technology that is affordable for these SMME to use, through the utility approach to service delivery [3]. The GUISET infrastructure allows these SMME to share information by helping them market and sell their products without spending much on the technology. The infrastructure allows these SMME to subscribe for only needed services that are available in the GUISET GRID and that can help in the marketing and selling of the products online. This again raises the issue of security concerns since during an online transaction or payment, sensitive data is being processed, and this then creates a need for such data to be properly secured [3].

II. REQUIREMENT ANALYSIS

In this section, we mainly focus on the analysis of requirements that leads to the design of the authorization framework and the model of authorization as applicable in the GUISET environment. We also present a typical usage scenario and some important security requirements as well as other vital design considerations for the GUISET infrastructure.

The analysis is based on the requirement analysis for GUISET authorization framework and on the collection of all relevant and critical information pertaining to the framework.

a) **Requirement name: Resource Request**
Description: This feature allows the client to make a request for the service that he/she want to access.
Justification: The framework should evaluate client request against the policies before granting GUISET services.

b) **Requirement name: Grant/Deny Access**
Description: The framework should ensure that every GUISET resource is secured from being access by unauthorized client by using policies.
Justification: The framework should allow authorized client to have access to the GUISET services that are ready available.

c) **Requirement name: Decision**
Description: This feature allows the framework to make decision based on the client request for the services.
Justification: The framework should make sure that client must conform to service policy in order to have to have access to the service.

III. DESIGN METHOD

A. USE CASE DIAGRAM

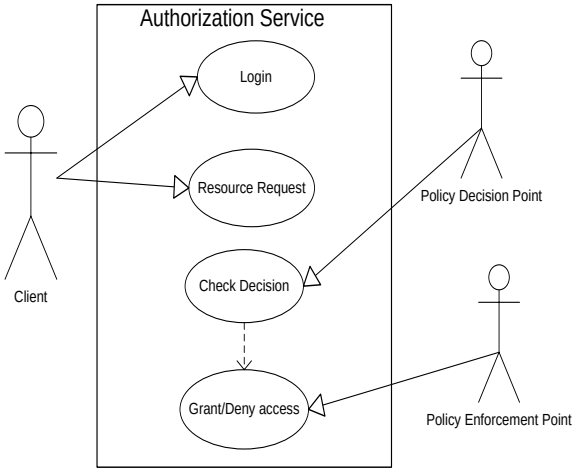


Figure1: Use Case Diagram

B. Use Case Descriptions

Use case descriptions are as follows

i) **Login Details**

TABLE I: LOGIN DETAILS

Use Case Name	Login.
Participating Actor	Initiated by Client communicating with the system.
Entry Condition	The client login to access information on the system.
Flow of Events	Client enters username and password to access information on the system.
Exit Condition	Opens a new window if the username and password is correct.

ii) Resource Request

TABLE II. RESOURCE REQUEST

Use Case Name	Resource Request
Participating Actor	Initiated by Client.
Entry Condition	Client makes a resource request to the system.
Flow of Events	The controller of a resource or service-a Policy Enforcement Point (PEP) - checks whether the client Authorized to have access to the resource or with the service with the Policy Decision Point (PDP).
	The PDP provide authorization service to the PEP
Exit Condition	PEP check decision provide by PDP to either grant/deny client access.

iii) Check Decision

TABLE III. CHECK DECISION

Use Case Name	Check Decision.
Participating Actor	Initiated by PDP.
Entry Condition	PDP check the message that was pass by PEP based on the client request
Flow of Event	The PDP provide authorization service to the PEP.
Exit Condition	The PEP check decision if client is authorized to have access to the resource.

iv) Deny/Grant Access

TABLE IV. DENY/GRANT ACCESS

Use Case Name	Deny/Grant Access
Participating Actor	Initiated by PEP.
Entry Condition	PEP check the decision that was made by PDP based on the client request to the service or resource.
Flow of Event	PEP check whether the client is authorized to have access to the service or resource with the PDP
Exit Condition	PEP check permission of client, if client is authorized then PEP grant access, if client is not authorized PEP deny access.

C. GUISET Authorization Scenarios

GUISET Authorization scenario examples are:

a) GUISET Scenario Name: Accessing new products
Participating Actor: Khulani Ngwenya
Flow of Events: Khulani as GUISET client submit his credential to the BBC Company marking a request for accessing new products as a client to the company.
: The BBC Company checks whether Khulani is an authorized registered client to the company who can access the new products that are available.
: If Khulani have been granted access and proven to be an authorized registered client in the company then Khulani can access new product that are in the company.

b) GUISET Scenario Name : Search for product
Participating Actor: Khulani Ngwenya
Flow of Events: Khulani as client of GUISET Infrastructure he wants to search for new product in BBC Company of his choice.
: Khulani as an authorized client in the GUISET Infrastructure he can now search for the product of his choice as an authorized client.

IV. GUISET ARCHITECTURE

The architecture is 3 tiers consisting of multi-modal interfaces, middleware layer and the GRID infrastructure layer. Each tier consists of its relevant components. In the GUISET architecture, we pay more attention on the GRID Infrastructure Layer as this is where services or resources reside and therefore there is a great need to ensure that only authorized Client has access to these services [4].

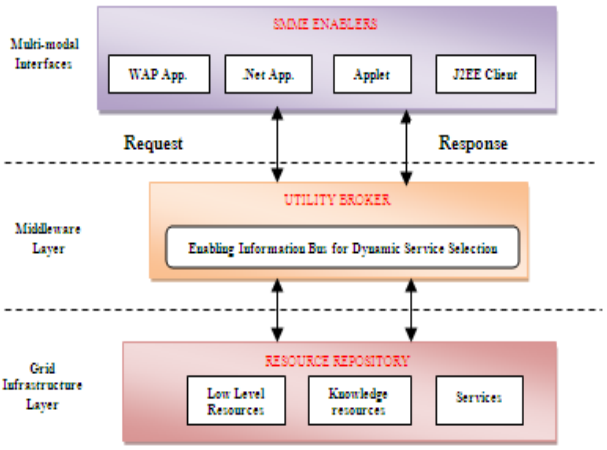


Figure 2. GUISET ARCHITECTURE

V. MESSAGE BROKER COMPONENT

This message broker component is the intermediary in the access path between the Subject and the web service operation or the resource being requested, it manages all the interactions related to authorization and access control by intercepting all SOAP messages when requests to services are being made [6].

This component not only controls access to web service requests but also all interactions related to the collection of authorization information. The figure below depicts how messages are intercepted before access to a resource is granted [6].

The UML sequence diagram, Figure 4 shows the sequence related to the message broker component, it shows the interaction of the various sub-components of the message broker for a basic operation [6].

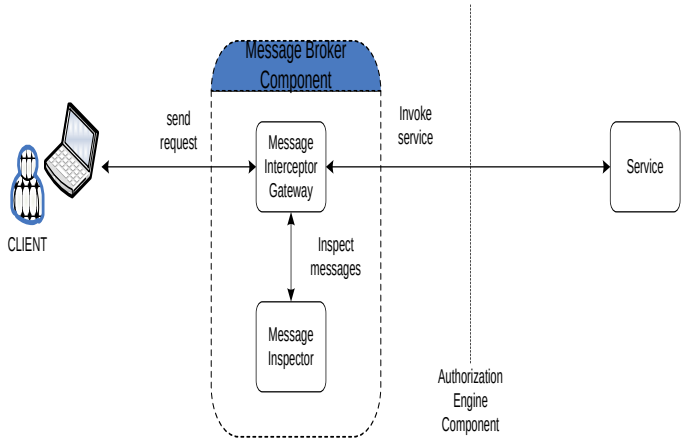


Figure 3. Message Broker component

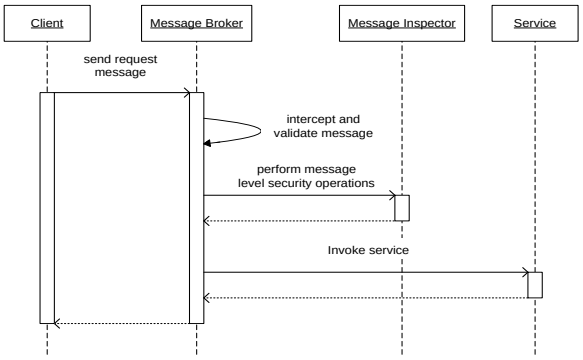


Figure 4. UML sequence diagram – Message Broker component

VI. GUISET AUTHORIZATION FRAMEWORK

Authorization in a distributed environment such as GRID needs to be flexible and scalable to support multiple security policies [6]; we made use of the XACML (eXtensible Access Control Markup Language) and SAML (Secure Assertion Markup Language), which are the two recognized important authorization related standards [1, 6].

As shown in Figure 5, authorizing a request from subject is first intercepted by PEP (Policy Enforcement Point).PEP constructs an authorization decision query and passes it to authorization handler. The result of this query determines if the request is to be granted/deny access to requested Service/Resource. The authorization decision query has details of about the identity of the subject, the service requested and the purpose for which service is requested [1].

Authorization Engine passes this information to PDP (Policy Decision Point). The policy is retrieved by PDP from PRP (Policy Retrieval Point). If the policy information is not available at PRP, it may be retrieved from policy store. The policies are written by administration using PAP (Policy Administration Point). PIP (Policy Information Point) is used by authorization engine to retrieve attributes of resources, subject and environment. After retrieving this information, authorization engine prepares a final result and passes it to PEP. If subject conforms to established privacy and other policies, PEP grants access of service/resource to subject, otherwise the access is denied [1].

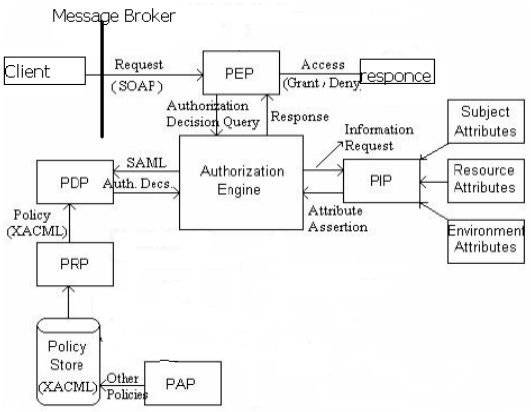


Figure 5. Authorization Framework for GUISET

VII. AUTHORIZATION MODEL

Authorization Engine Component

The authorization model must be able to support multiple security policies and need to have the flexibility to support dynamic changes in security policies [4]**.** The Authorization Engine component is responsible for making decision based on the authorization policies and access control policies stored in the policy repository, these ensure that authorized client requesting for services are only granted access to services/resources they are only requesting. Every Service Provider within a Domain has its own policies and he can change them dynamically [1].

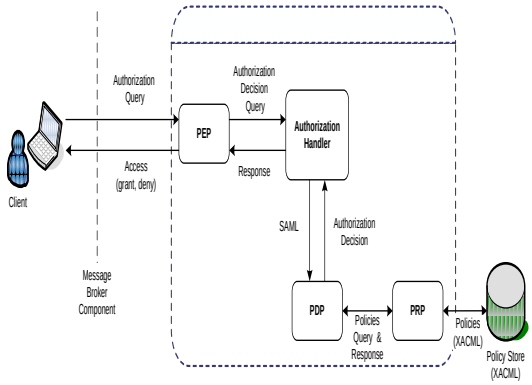


Figure 6. Authorization Engine component

Figure 6 shows a typical application environment where a user would want to access a particular service or resource, the authorization decision request from the Message Broker component is first intercepted by Policy Enforcement Point (PEP), this constructs an authorization decision query which contains information on the acquired token of the service requestor and also the details of the service being requested. This basically validates a client whether to access the resource being requested, and then this query is passed onto the Authorization Handler (AH) which then passes this information to the Policy Enforcement

 ISBN: 978-1-61208-106-9

Point (PDP), which then looks up the security access control policies [1].

The results of this query determine whether access to resource/service is granted. The PDP responsibility is to retrieve policies from the policy store using a Policy Retrieval Point (PRP) and if the policy is not available in the PRP, it may then be retrieved from the Policy Store, the Policy Store has capabilities of importing/exporting these policies in an XACML form and are constructed as a set of rules against the target service, i.e., (Client, resource, action) Client refers to the requestor, resource refers to the service or resource being requested and action refers to the kind of action to be performed on the requested service [1].

The significance of using XACML is due to the fact that it provides both a policy language and an access-control decision request/response language to meet the security access control requirements. With XACML, the PEP forms a query language to ask the PDP whether or not a given action should be allowed. The Authorization Engine get information from the PDP and after that the Authorization Handler prepares the final results then passes it to the Policy Enforcement Point which then passes it to the Message Broker component, based on these results the Policy Enforcement Point then returns a value of either (grant or deny) access to the requested resource [1].

VIII. IMPLEMENTATION

The main focus of this section is to show how clients interact with the system and how an authorized client accesses the services that are provided in the GUISET. We use diagrams to show how client interacts with the system before granting a service.

A. GUISET Authorization

One may want to ask how a client is granted an access for a service. This is easily achieved through the following steps:

a) Receive the data request and authentication tokens of the client that is requesting for service.

b) After receiving data request, make an authorization request for the service to the authorization engine.

c) The authorization engine will then return the authorization decision, and based on that decision, it will then know whether the client is authorized to access the services or not.

The above mentioned steps will now be presented through pictorial narration of client user interfaces.

B. GUISET Main page

This is the main page of the GUISET and it is the entry point to the GUISET Infrastructure. The main page of a GUISET has a Login button and register button. When a client clicks the login button, a Login dialogue appears.

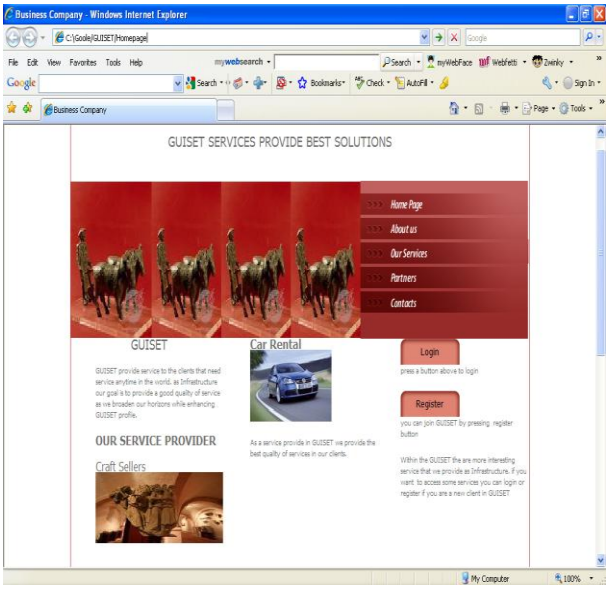


Figure 7. GUISET Main page

C. THE RESULT OF IMPLEMENTATION:

Figure 8 shows how authorization takes place in GUISET when a client wants to access services that are available.

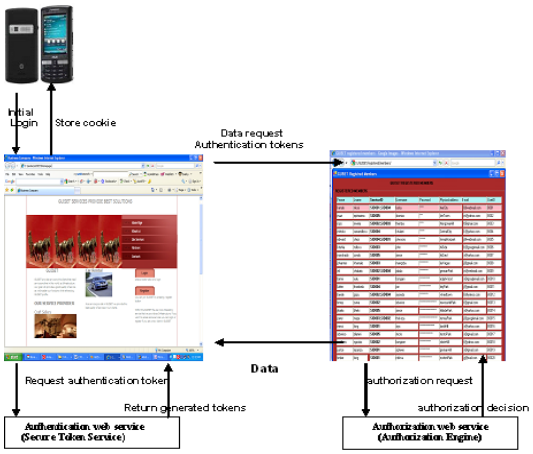


Figure 8: Result of Implementation

IX. CONCLUSION

Grid computing has become a very useful and interesting approach to enhance solution provisioning, sharing of resources, data and services in a distributed manner. The provision of large scale distributed services requires high level of authentication and authorization for access control. Available methods are not suitable to our GUISET environment as they seem to be dynamic to the environment of operation. GUISET with its dynamic nature also, and as part of our research focus, requires its framework and

 ISBN: 978-1-61208-106-9

implementation for its operation. Consequently, as reported in this paper, we have designed and developed a framework whose contribution is geared towards acting as a gatekeeper for access control in our GUISET service provisioning environment.

REFERENCES

[1] S. Singh. and S. Bawa, "A Privacy, Trust and Policy-based Authorization framework for Services in Distributed Environments". International Journal of Computer Science, Volume 2 Number 2 2007, ISSN 1306-4428, pp 85-92

[2] L. Marchel, et al., "Optimal Bandwidth Sharing in GRID Environment," Proceedings of the 15th IEEE International Symposium on High Performance Distributed Computing, HPDC-15, Paris, France, June 19-23, 2006, pp 153-163.

[3] T. Ziebermayr and S. Probst, "Web Service Authorization Framework." IEEE International Conference on Web Service (ICWS'04), San Diego, California, June 6 -9, 2004, pp.67-74

[4] M. Adigun.; J. Emuoyibofarhe and O. Migiro, "Challenges to Access an Opportunity to use SMME enabling technology in Africa". 1st ALL African Diffusion Conference, Johannesburg, South Africa, June 12-14, 2006, pp. 34-43.

[5] I. Foster,et al.,"A Multipolicy Authorization Framework for GRID Security". Proceedings of the 5th International Symposium on Network Computing and Applications. IEEE Computer and Society, Washington, DC, USA, July 2006, pp 269-272.

[6] R. Tatyana, N. Clifford, and L. Zhou, "Integrated Access Control and Intrusion Detection (IACID) Framework for Secure Grid Computing". Technical Report, USC Internet and GRID Computing Lab (TR 2004-6) May 21, 2004

 ISBN: 978-1-61208-106-9

Parallelization Programming Techniques: Benefits and Drawbacks

Goran Martinovic
Faculty of Electrical Engineering
J.J. Strossmayer Univ. of Osijek
Osijek, Croatia
e-mail: goran.martinovic@etfos.hr

Zdravko Krpic
Faculty of Electrical Engineering
J.J. Strossmayer Univ. of Osijek
Osijek, Croatia
e-mail: zdravko.krpic@etfos.hr

Snjezana Rimac-Drlje
Faculty of Electrical Engineering
J.J. Strossmayer Univ. of Osijek
Osijek, Croatia
e-mail: snjezana.rimac@etfos.hr

***Abstract*-In engineering program implementations, there is always a need for more computer resources, apart from which, many computer resources are still unused. The most common resource demanding applications are applications which require a lot of processor power or RAM space. Today, advancing technology offers processing power in grids, clusters, multi-core CPUs, cloud computers, or even graphics processing units. Thus, given all this computing power, smart and efficient utilization of these systems is needed. All these necessities and mentioned facts laid foundation for parallel programming. One of the major issues in parallel programming is reconfiguring of the existing applications to work on a parallel system; not just to work, but to work faster and more efficiently. In this paper some of the most common parallelizing methods will be presented using MPI on the Croatian National Grid Infrastructure (CRO-NGI), as well as their advantages in terms of cost-effectiveness and simplicity.**

Keywords: computational grid, load balancing, MPI, parallel computing.

I. INTRODUCTION

The complexity, data requirements and processing in scientific researches, such as visualization and modeling in various scientific branches continue to increase. Problems in medicine, weather prediction, global climate modeling, complex stress calculations in mechanics, etc. are good examples of computer intensive applications. Historically, the computational power of computer resources has not been able to keep pace with this increase and for this reason, parallel computer systems (PCS) were developed. Not every resource intensive problem can be solved in decent time manner on simple mainstream computers, as shown in [3] and [10]. Single-processor systems and single core processor computers by themselves are getting time consuming in running these applications, and are causing major drawbacks of developing such applications. As resource consumption by computers has become a concern in recent years, parallel computing has become the dominant paradigm in computer architecture, mainly in the form of multi-core processors [14]. Today, there are various types of parallel computing systems, like clusters, grids, distributed systems, multi-core and many-core processors and a recent concept – cloud computing systems, all based on spreading use of parallel algorithms. Nowadays parallel computers are very common in research facilities as well as companies all over the world and they are used extensively for complex computations. Some of the most powerful supercomputers are made with over 10,000 processors, and are capable of reaching over 1 petaFLOPS. Careful and effective parallel programming is the only way to bridle such enormous computing power. Massive migration to parallel systems causes that still many applications need to be adjusted for the use on these systems by means of two main options: recoding or writing the code in parallel from scratch. Sometimes, these procedures are not as intuitive as one may think, because not all tasks can be parallelized.

In this paper different parallelizing methods and techniques will be mentioned. A couple of experiments will be used to support or to undermine these aforementioned facts and myths about parallel programming. Finally, efficiency of parallel computer systems will be questioned, and their advantages and disadvantages will be compared to serial systems. Section 2 describes basic aspects of parallelization. The concept of automatic parallelization is briefly explained in Section 3. Prior to experimental results given in Section 5, Section 4 presents formerly known benefits of parallel programming in order to experimentally confirm or reject them. Section 6 presents planned work, and Section 7 brings noted conclusions.

II. PARALLELIZATION METHOD IN USE

Many parallelization tools and compilers are already available and yet many more are in development. Two basic modes of parallelization are automatic parallelization and manual parallelization. Automatic parallelization techniques are mainly the tools for real-time systems and scientific computing industry [1], while manual parallelization allows maximum application optimization for parallel execution and performances gain. Interesting concepts are parallelization compilers, which turn codes parallel and make the application parallel at runtime by which they can be perceived as automatic parallelization techniques [12]. Manual parallelization techniques are issued by programmers. The latter techniques are based on speculative parallelization [8], parallelization of loops [16], dynamic data parallelization [2], load balancing, thread pipelining,

 ISBN: 978-1-61208-106-9

data access partitioning [7] and others. Parallel programming techniques, parallel platforms and parallel programming tools are in constant development.

The main goal of this paper is to point out some problems that appear by optimization of sequentially written programs for execution on parallel platforms, and presenting various parallelization methods. Various data structures, loops and iterations can be parallelized efficiently without rewriting the whole program code. Another important issue is load balancing, the goal of which is to gain a higher throughput, and to reduce the user-perceived latency, especially in the case of high network traffic or a high request rate causing the network to be bottlenecked, or a high computational load, [11]. Load balancing is hardly achieved on, for example, cloud computing systems, because of the high system heterogeneity, especially network. These systems are overwhelmed with inconsistency and are prone to many changes in the matter of seconds, and that is why these systems have to use standalone load balancing appliances or content switches.

The first step of parallelizing an application is to find the most resource/time intensive part of the program (algorithm). If this part cannot be made parallel, little can be done for speeding up the application. The next step, step two, is to determine whereas parallel parts of the code are data independent, and remove this dependency if possible. Then there comes step three, i.e., determining a method which will be used in parallelization of kernels. These steps alone comprise several sub-steps, which are chosen according to the problem at hand. Several concepts will be introduced in this paper for parallelization: data partitioning, loop parallelization and functional decomposition, which are given in detail in [2] and [9]. Other concepts are briefly presented in Table 1.

Data parallelization (or data decomposition) is based on parallelizing data, e.g., dividing large databases, matrices, vectors and other data types into small chops often adjusted to be processed on the nodes of parallel systems. Data can be divided equally or in some other manner (load balancing or adaptive parallelization, [11]). A negative impact of dividing data in that manner is conspicuous on computer grid systems with nodes interconnected with the network, because the network is the slowest subsystem in inter-application communication. Other important issue to be confronted with is reducing communication between processes. There are three different types of algorithms based on communication frequency: coarse-grained, fine-grained and embarrassingly parallel. There are various methods of reducing user perceived latencies, which are derived from inter-process communication. Another shortcoming can be seen in heterogeneous systems in which data should be divided in accordance with available computer resources, see [10]. In heterogeneous systems the best performance gain would be noticed if there is a system monitoring current resource availability and sending information about available resources back to the application, which in turn sends appropriate portions of data to computer nodes, as shown in [10]. These systems exist in cloud computing systems, however, with only limited functionality. There is a number of possible data partitioning, first when portions of data are sent to nodes, and second when a copy of the whole data is sent to all computer nodes, as in Figure 1, the latter using more communication, so it has to be tested thoroughly which system benefits from this approach. Figure 2 shows which data are computed by which node.

```
myRank = MPI::COMM_WORLD.Get_rank();
nProc = MPI::COMM_WORLD.Get_Size();

for(i=((CONST/nProc)*myRank);i<(CONST/nProc)+(
(CONST/nProc)*myRank);i++) {
   for (j=0;j<DIMV;j++) {
      if ((maxi[11] < optiMatrix[i][j]) &&
(matrix[i][j][16] !=0)) {
         maxi [0] = I;
         maxi[3] = j;
         maxi [11] = optiMatrix [i][j];
      }
   }
}
```

Figure 1. Example of data parallelization

Node A	Node B	Node C

A	B	C	D	E
F	G	H	I	J
K	L	M	N	O
P	Q	R	S	T

Figure 2. Visual representation of data division amongst parallel nodes

Dividing data in programming is tightly coupled with loop parallelization, because data containers are generally accessed by loops, so a condition deciding which data is going to which node should be put in the loop initialization and termination. Loops are often parallelized by dividing the number of iterations equally, if possible, amongst computer nodes, as in [3], [9], [13], and [16]. Data division is prone to data dependency, and by that care must be taken when preparing data for parallel distribution. Loops can be made parallel only if iterations are not data dependent. In other words, Bernstein's conditions [4] have to be fulfilled.

The third concept is based on functional decomposition, described in [4] and [9]. This is done by dividing the program into functional blocks independent of each other at the time of execution, e.g., one does not require data of the other. This is explained by the Church-Rosser property [4], which holds that the arguments to a pure function can be evaluated in any order or in parallel, without changing the result. A negative side of functional decomposition can be seen if parallel computations differ in the execution time a lot, because the slowest one determines the total execution time of this parallel part assuming that a computation result must be provided before processes move on to the next

 ISBN: 978-1-61208-106-9

program block, except in applications with active load balancing. If the latter is not the case (computation results are processed independently), the algorithm is called *embarrassingly parallel*. Embarrassingly parallel algorithms are often used in cloud computing systems.

Main program			
Declarations and initializations Common procedures			
Node 1	**Node 2**	**…**	**Node n**
Search the DATA for value "A"	Search the DATA for value "B"	…	Search the DATA for value "N"

Figure 3. Example of a pseudo-code of data parallelization for searching for different values in the same data set

A living example is searching for two or more values in a data matrix, then the first node searches for one number and the second node searches for the other (if all nodes are assumed to have access to all data), as in Figure 3.

Computer nodes intercommunication is generally done by sending some signal in a preconfigured way by using some routines. This type of parallel programming is called Message Passing. Message passing applications communicate over a high speed network in distributed computing systems, or over high speed buses in shared memory computers, so the program can hold sufficient cohesiveness.

III. AUTOMATIC PARALLELIZATION

Given the example of [1], [3], [5] and [14], in order to remove the burden from a programmer to manually rewrite sequential codes for parallel execution, many new methods are introduced as an attempt to solve this problem automatically. They often comprise compilers which "know" how to parallelize a certain program code. A vast majority of automatic parallelization compilers are developed for FORTRAN, such as the Vienna Fortran compiler, the Paradigm compiler, the Polaris compiler, the SUIF compiler, and some of the concepts independent of the programming language, such as commutativity analysis [14]. Automatic parallelization is the ultimate goal for parallel programming, as it removes the programmer from the parallelizing part in coding the application, thus making parallel applications production faster and more efficient. Every automatic parallelization concept has been done only with limited success. Despite poor progress, automatic parallelization has been intensively researched for the past few decades, and a lot of work is still dedicated to it.

An interesting concept for automatic parallelization is presented in [14], which is called commutativity analysis. It aggregates both data and computation into larger grain units. It then analyzes computation at this granularity to discover when pieces of computation commute (i.e., generate the same result regardless of the order in which they are executed). If all of the operations required to perform a given computation commute, the compiler can automatically generate a parallel code. Some sources also describe various hybrid approaches, such as in [9]. Parallelization methodologies are expanded in Table 1, which gives additional information about other main parallelization techniques, advantages and disadvantages. Even with many methods for automatic parallelization, fully automatic parallelization of sequential programs by compilers remains a grand challenge due to its need for a complex program analysis and the unknown factors (such as the input data range) during compilation. Automatic parallelization combined with cloud computing systems in near future will probably serve as self-sufficient parallel systems, which will bring high performance computing to every computer user connected to web.

IV. BENEFITS OF PARALLEL PROGRAMMING

It would be expected of parallel programs to have the execution time cut in proportion with the number of nodes, as opposed to sequential programs. However, if this fact is analyzed more thoroughly, there is always some portion of code which cannot be parallelized, and that portion must be taken into account. This issue was addressed by Amdahl's and Gustafson's law [3]. Some researchers noticed that even with Gustafson's law, which suggests that it is beneficial to build a large-scale parallel system as the speedup can grow linearly with the system size, there are some physical constraints which do not allow many applications to scale up and meet the time bound constraint. In practice, that constraint is often of physical nature in the form of memory limitation.

To sum it all up, in [15] authors propose a memory bounded speedup model. Amdahl's law is a special case of a *memory bounded speedup model*. This model is greatly applicable to multi-core systems and GPU cores, which represent home shared memory systems. Embarrassingly parallel applications are not affected by Amdahl's law, at least not to such a great extent, and that is why these applications can easily be run on clouds. These applications often comprise independent tasks, which are then executed on different computer nodes, without a need for any type of communication, or data dependencies, except at the beginning of a code. With all added up, parallel applications have many benefits from today's parallel systems. Most of parallelizing methods can be run on almost all existing parallel systems. There are limitations, but with constant research in this field, the number of limitations decreases. For example, cloud computing systems can serve as parallel platform only for applications that are easily parallelized. In all other cases this is nearly impossible, because cloud system is too hard to handle resource-wise, which is caused by the high system heterogeneity. With parallel system properties in mind, it is easy to classify given parallelizing methods from Table 1 in correspondence with these platforms. Every existing application can be more or less parallelized; it is the question of cost and time-effectiveness, a parallel system on which the application is run, time bound constraints on application and application environment which method will be used.

 ISBN: 978-1-61208-106-9

V. EXPERIMENTAL SETUP

In order to visualize given facts into real appliances benefits and speedup, two experiments will be shown. In the first experiment, there are two multi-criteria optimization algorithms, whose performance will be compared. In the second algorithm, a parallel image processing algorithm is tested in different environments and with a different setup.

A. Experiment 1: Multi-criteria optimization algorithms (PMCO1 and PMCO2)

The first algorithm, Parallel Multi-Criteria Optimization 1 (PMCO1) is an example of computing large data in parallel with a small amount of communication between processes. It uses the approach described in [10]. The system makes a decision based on different preferences amongst options in a large data set. There are databases containing various system parameters, and prior to program execution it is necessary to extract data from these databases, which are formed by plain text files. PMCO1 reads different databases in different computer nodes.

TABLE I. PARALLELIZATION MODELS COMPARED

Parallel Program Design	Parallelization technique	Positive features	Negative features
Manual Parallelization	**Shared Memory**	No need for communication between tasks	Difficult data locality management
	Threads	Fine program granularity and efficient platform utilization	Not reusable, errors affect whole process
	Data Parallel Techniques	Large performance increase, error on one data "chunk" rarely affect other data	No performance increase if the data is not independent, need for task communication
	Message Passing	Universality, Data locality management, Easy debugging	Programmer manages memory placement and communication occurrence
	Hybrid	Combination of the techniques above	Combination of two or more parallelization techniques can greatly reduce their disadvantages
Automatic Parallelization i.e. parallelizing compilers (pre-processors)	**Fully Automatic**	Parallelization without programmer, fast parallel code generation, computer aided parallelization cost-effectiveness analysis	Can produce wrong results, application performance can be actually degraded, much less flexible than manual techniques, if the code is too complex parallelization cannot occur
	Programmer Directed	Usage of compiler directives, better parallelization management	

B. Experiment 1: Multi-criteria optimization algorithms (PMCO1 and PMCO2)

The first algorithm, Parallel Multi-Criteria Optimization 1 (PMCO1) is an example of computing large data in parallel with a small amount of communication between processes. It uses the approach described in [10]. The system makes a decision based on different preferences amongst options in a large data set. There are databases containing various system parameters, and prior to program execution it is necessary to extract data from these databases, which are formed by plain text files. PMCO1 reads different databases in different computer nodes. This parallelism is based on function level parallelism (FLP). The second algorithm, Parallel Multi-Criteria Optimization 2 (PMCO2) is a parallel algorithm which reads all databases in every node, and serves the analysis of the computational part of the program. PMCO2 is mainly a data parallel model, but it uses a hybrid approach, explained in [2], comprising FLP and data parallelism, and for the purpose of illustration, its execution time is divided into reading data and computation. In PMCO2, every node has access to all data and there is a significant process communication time, but communication takes place rarely. PMCO2 approach enables all nodes to read only a portion of data, regardless of the fact that they contain the whole database. On the other hand, PMCO1 has more frequent communications between processes, because nodes contain only a portion of data. These small communications can make a great deal if the database is very large as in Table 3. In a small data set, PMCO1 tends to have better performance (Table 2). PMCO2, though, has one more drawback, which is memory-wise, considering the fact that every node holds all data. So if the data is too large, it would not be possible to run the program based on PMCO2, whereas on PMCO1 it would be possible but slow.

C. Experiment 2: Parallel Image Processing Algorithm (PIPA)

The second experiment deals with an image processing algorithm, which does some basic pixel manipulation on the grayscale satellite images in different sizes. The application was run in parallel on the Croatian National GRID infrastructure on 4 and 8 nodes, as well as on two different CRO-NGI installations (ETFOS, located at the Faculty of Electrical Engineering, University of Osijek and SRCE (University Computing Center in Zagreb). What can be easily seen is the difference in performance, as well as the impact of different architectures and operating systems on the application execution time. The application was also run on a PC with a dual-core processor. In that way it can be analyzed whether parallel programming and execution on parallel systems can be justified by taking performance into main consideration. Figure 4 shows dependency of the application execution time on image size by different operating systems, numbers of nodes, numbers of processor cores and image sizes.

 ISBN: 978-1-61208-106-9

TABLE II. ALGORITHMS PMCO 1 AND 2 PERFORMANCE TEST (A SMALL DATA SET)

		4 – node computer grid time (s)	6 – node computer grid time (s)
PMCO 1	Data read	1.79	0.78
	Computation	3.77	8.00
	Total	5.56	8.78
PMCO 2	Data read	12.24	15.46
	Computation	0.20	0.15
	Total	12.44	15.61
Program execution time difference (s)		-6.88	-6.83

TABLE III. ALGORITHMS PMCO 1 AND 2 PERFORMANCE TEST (A LARGE DATA SET)

		4 – node computer grid time (s)	6 – node computer grid time (s)
PMCO 1	Data read	9.89	3.18
	Computation	261.67	206.27
	Total	271.56	209.44
PMCO 2	Data read	42.60	55.75
	Computation	24.21	17.23
	Total	66.81	72.98
Program execution time difference (s)		204.75	136.46

Image size affects performance most, which is expected, because image size grows almost exponentially.

Also, the number of grid nodes and processor cores, which can be distinguished in Figures 5 and 6, has a great impact on performance. Figure 7 shows that careful multi-core parallel programming can lead to a significant performance boost, which is almost 90% of the increase, with doubling the number of cores. Multi-core processors show their true strength when loaded with applications optimized for multi-core execution. Also, multi-core platforms do not have one major drawback which grids and clusters have, and that is a relatively large process communication time in message passing applications. Figure 9 shows a process communication impact on application performance. It clarifies what was mentioned before; i.e., the grid suffers from great performance loss when using too much of communication between processes. Furthermore, this is more expressive in public computational grids and cloud systems, whose networks are always under some load, leading an application expected to finish faster to finish slower, waiting for processes to finish their communication. On the other hand, there is a communication between processes run on multiple cores onto one processor, whose process communication time can be safely ignored in performance analysis. This issue is a problem of its own and part of future work based on heterogeneity modeling. But not to be confused, grids offer a big advantage compared to multi-core processors. They can have much more nodes, which in turn can have multi-core processors themselves, and modern commercial multi-core processors can have only up to 8 cores, so it is up to the application which platform should be used in its execution. Another example are clouds, which are hybrid parallel systems and offer various performance advantages.

More performance boost can be obtained by increasing the number of computer nodes executing the application, as shown in Figures 4 to 7. Usage of ETFOS installation of CRO-NGI lowers process communication a bit (because of a lighter network load), and decreases execution time significantly.

VI. FUTURE WORK

Further work will be based on implementing more parallel platforms such as GPGPUs (General-purpose computing on graphics processing units) in NVidia CUDA (Compute Unified Device Architecture) and ATI Stream [6]. This work will tend to put these parallel newcomers into performance tables of other parallel systems. Another research is covering the role of cloud computing systems as parallel systems with great computing power. Image processing algorithms will be thoroughly reworked. Due to image size limitations, new image processing algorithms will be used with support for color images. There are other algorithms being developed; each of them will put different aspects of parallel systems onto test. There is also an idea to provide a mathematical proof for the exact limit of parallelization efficiency, and profitability of using parallel systems opposed to other parallel and non-parallel systems.

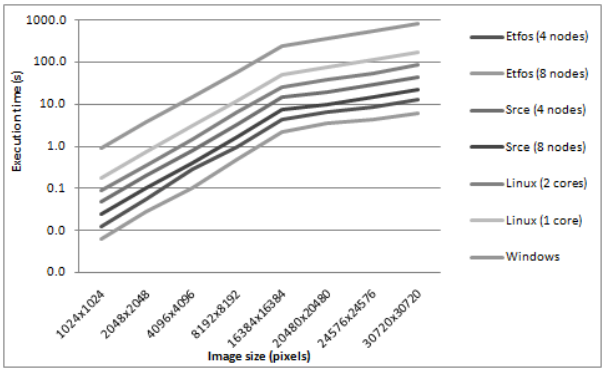


Figure 4. Performance gain

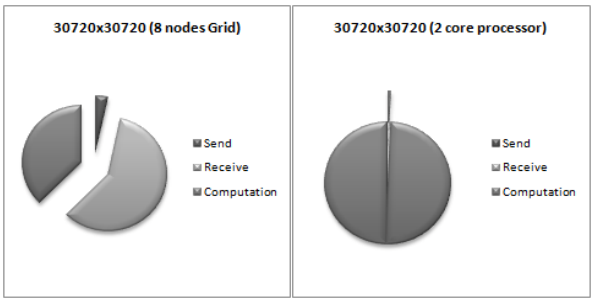


Figure 5. Communication impact on the overall performance

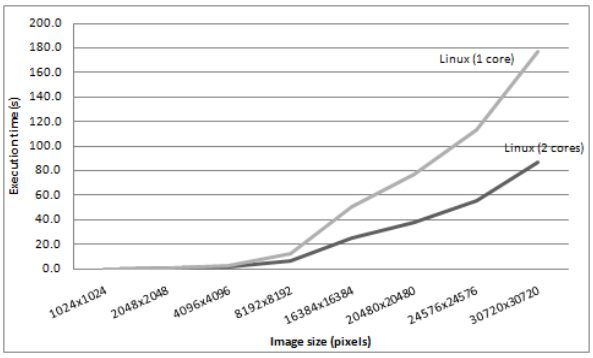


Figure 6. Performance with a different number of processor cores

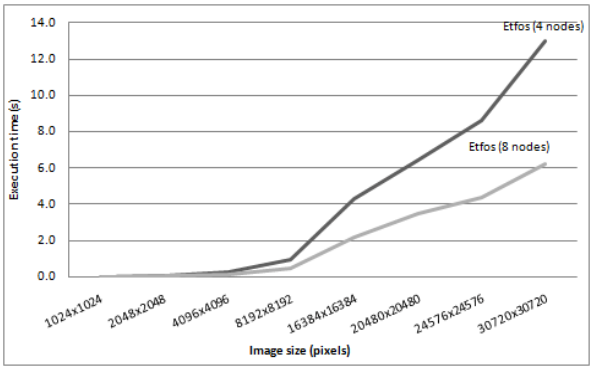


Figure 7. Performance with a different number of computer nodes

VII. CONCLUSION

The latest technologies give many opportunities when it comes to execution of demanding applications. More recent parallel systems such as computational grids, clusters, multi-core systems, Massively Parallel Processors systems and Graphics Processing Units are platforms that offer much more computer resources than standard PCs, and their ideas and technologies are slowly making their way to desktop computers. The best examples are multi-core computers, which share some backbone principles with parallel systems. Many existing applications are made sequential, but sometimes with just few changes they can be made parallel, therefore reducing their executing time and other time demands. Parallelizing methods are chosen in accordance with the type of parallel systems they are run on. Many parallelizing methods are presented today, and their ultimate goal is to produce a fully automatic parallelizing system, which can be used as any other today's programming language and system. The aforepresented experiments prove that parallel programming is the present and the future of all scientific research, not only in computer science, but in other researches as well. Parallel programming brings an enormous performance advantage, possibilities such as ability to process larger data and to calculate more complex mathematics and statistics. But this is only possible if these systems, their potential and their drawbacks are understood well. In order to make use of squeezing most out of these systems, careful and wise resource usage is needed, as well as efficient programming and work distribution. There are many other parallelization concepts, many of which are still being in their infant phase, but there is no doubt that research involved in them will produce more efficient, scalable and simple parallel applications and mechanisms. Such techniques involve function and block level parallelization, automatic parallelization, parallelization techniques for heterogeneous systems, speculative parallelization models, loop based parallelization techniques, and many more. Undoubtedly, parallel applications bring more performance, more options, and remove barriers for many research branches, bringing the overall sky-high progress in computing technology. It is obvious that in future, by combining these and similar methods, most advanced parallelization techniques will be created and parallel computing will be driven into the mainstream.

ACKNOWLEDGMENT

This work was supported by research project grant No. 165-0362980-2002 from the Ministry of Science, Education and Sports of the Republic of Croatia.

REFERENCES

[1] B. Armstrong and R. Eigenmann, Application of Automatic Parallelization to Modern Challenges of Scientific Computing Industries, Proc. 2008 Int. Conf. Parallel Processing, Portland, OR, USA, Sept. 8-12, 2008, pp. 279-286.

[2] D. Banerjee and J. C. Browne, Complete Parallelization of Computations: Integration of Data Partitioning and Functional Parallelism for Dynamic Data Structures, Proc. 10th Int. Parallel Processing Symp., Honolulu, HI, USA, Apr. 15-19, 1996, pp. 354-360.

[3] U. Banerjee, R. Eigenmann, A. Nicolau, and D.A. Padua, Automatic Program Parallelization, Proc. of the IEEE, Vol. 81, No. 2, 1993, pp. 211-243.

[4] J. Błażewicz, K. Ecker, B. Plateau, and D. Trystram, Handbook on Parallel and Distributed Processing, Springer, 2000, pp. 96-173.

[5] U. Bondhugula, et al., Towards Effective Automatic Parallelization for Multi-core Systems, Proc. 22nd IEEE Int.

 ISBN: 978-1-61208-106-9

Symp. Parallel and Distributed Processing, Miami, FL, USA, Apr. 14-18, 2008, pp. 1-5.

[6] G. Chen, G. Li, S. Pei, and B. Wu, High Performance Computing Via a GPU, Proc. IEEE Int. Conf. on Information Science and Engineering, Shanghai, China, Dec 26-28, 2009, pp. 238 - 241.

[7] M. Chu, R. Ravindran, and S. Mahlke, Data Access Partitioning for Fine-grain Parallelism on Multi-core Archs., Proc. IEEE/ACM Int. Symp. on Microarchitectures, Chicago, IL, USA, Dec. 1-5, 2007, pp. 369-380.

[8] C. Tian, M. Feng, V. Nagarajan, and R. Gupta, Copy or Discard Execution Model for Speculative Parallelization on Multi-cores, 41st Ann IEEE/ACM Int. Symp. on Microarchitectures, Lake Como, Italy, Nov. 8-12, 2008, pp. 330-341.

[9] K. A. Kumar, A.K. Pappu, K.S. Kumar, and S. Sanyal, Hybrid Approach for Parallelization of Sequential Code with Function Level and Block Level Parallelization, Proc. IEEE Int. Symp. Parallel Computing in Electrical Engineering, Bialystok, Poland, Sept. 13-17, 2006, pp. 161-166.

[10] G. Martinović, L. Budin, and Ž. Hocenski, Static-Dynamic Mapping in Heterogeneous Comp. Environ., Proc. IEEE Symp. Virtual Environments, Human-Computer Interfaces and Measurement Systems, Lugano, Switzerland, July 27-29, 2003, pp. 32-37.

[11] G. Martinovic, Resource Management System for Computational Grid Building, IEEE Int. Conf. Systems, Man, and Cybernetics, San Antonio, TX, USA, Oct. 11-14, 2009, pp. 1312-1316.

[12] T. Nakatani and K. Ebcioglu, Making Compaction-Based Parallelization Affordable, IEEE Trans. Parallel Distributed Systems, Vol. 4, No. 9, 1993, pp. 1014-1029.

[13] V. Purnell, P. H. Corr, and P. Milligan, A Novel Approach to Loop Parallelization, Proc. IEEE Proc 23rd Euromicro Conf., Budapest, Hungary, Sept. 1-4, pp. 272-277.

[14] M. C. Rinard and P. Diniz, Commutativity Analysis: A New Technique for Automatically Parallelizing Serial Programs, 10th Int. Parallel Processing Symp., Honolulu, HI, USA, April 15-19, 1996, pp. 14-14.

[15] X.-H. Sun and Y. Chen, Reevaluating Amdahl's Law in the Multi-core Era, J. Parallel. Distrib. Comput., Vol. 70 (2010), pp. 183-188.

[16] C. Xu and V. Chaudhary, Time Stamp Algorithms for Runtime Parallelization of DOACROSS Loops with Dynamic Dependences, IEEE Trans. Parallel and Distributed Systems, Vol. 12, No. 5, 2001, pp. 433-450.

 ISBN: 978-1-61208-106-9

A Generalized MapReduce Approach for Efficient mining of Large data Sets in the GRID

Matthias Röhm, Matthias Grabert and Franz Schweiggert
Institute of Applied Information Processing
Ulm University
Ulm, Germany
matthias.roehm@uni-ulm.de, matthias.grabert@uni-ulm.de, franz.schweiggert@uni-ulm.de

***Abstract*—The growing computerization in modern academic and industrial sectors is generating huge volumes of electronic data. Data mining is considered the technology to extract knowledge from these data. With an ever increasing amount of data and complexity of modern data mining applications, the demand for resources is rising tremendously. Grid and Cloud technologies promise to meet the requirements of heterogeneous, large-scale and distributed data mining applications. The DataMiningGrid system was developed to address some of these issues and provide high performance and scalability, sophisticated support for different types of users, flexible extensibility features, and support of relevant standards. While the DataMiningGrid, like most of the related grid systems, focused on compute-intensive applications, Google's MapReduce paradigm and Cloud-Computing brought up new solutions for efficient data analysis. Based on the DataMiningGrid, we developed the DataMiningGrid-Divide&Conquer system that combines these important technologies into a general-purpose data mining system suited for the different aspects of today's data analysis challenges. The system forms the core of the Fleet Data Acquisition Miner for analyzing the data generated by the Daimler fuel cell vehicle fleet.**

***Keywords*-Data mining, Grid, MapReduce.**

I. INTRODUCTION

Increasing data volumes in many industrial and academic sectors are fueling the need for novel data analysis solutions. The effective and efficient management and transformation of these data into information and knowledge is considered a key requirement for success in knowledge-driven sectors. Data mining [1] is the key methodology to address these information needs through automated extraction of potentially useful information from large volumes of data. In the last decade there have been multitudes of efforts to scale data mining algorithms for solving more complex tasks, including peer-to-peer data mining, distributed data stream mining and parallel data mining.

Recently, data mining research and development has put a focus on highly data-intensive applications. Google's publications on MapReduce [2][3], a special incarnation of the Divide&Conquer paradigm, inspired many projects working on large data sets. MapReduce frameworks like Hadoop simplify the development and deployment of peta-scale data mining applications leveraging thousands of machines. MapReduce frameworks are highly scalable because they avoid data movement and rather send the algorithms to the data. In contrast to other data mining environments these frameworks restrict themselves to a certain programming model, loosing some of the functionality provided by fully featured queuing systems.

Another branch of modern distributed data mining is motivated by the sharing of heterogeneous, geographic distributed resources from multiple administrative domains to support global organizations. This field of active research and development is generally referred to as data mining in grid computing environments. The DataMiningGrid [4] project addresses the requirements of modern data mining application scenarios arising in grid environments, in particular those which involve sophisticated resource sharing. The DataMiningGrid system is a service-oriented, scalable, high performance computing system that supports grid interoperability standards and technology. It meets the needs of a wide range of users, who may flexibly and easily grid-enable existing data mining applications and develop new grip-based approaches. The DataMiningGrid, like most of the related grid systems, focused on compute-intensive applications leading to an architecture build on three components: (1) Specialized storage servers to store data and programs. (2) Compute clusters composed of multiple compute nodes for running the algorithms. (3) Grid management servers for managing the storage and compute resources connected to them.
In such an environment data is stored on dedicated storage servers and has to be transferred to the compute nodes prior to execution. Though different scheduling algorithms have been proposed to optimize the relation between data transfer and execution time, for data-intensive applications, data should not be moved at all [5].

To bring the advantages of the MapReduce paradigm into worldwide, heterogeneous computing environments we developed the DataMiningGrid-Divide&Conquer (DMG-DC) system based on the concepts and services of the DataMiningGrid project. This article is organized as follows:
First, we briefly revise MapReduce frameworks and introduce the more general Divide&Conquer paradigm for

 ISBN: 978-1-61208-106-9

data-intensive applications. Then we describe the DataMiningGrid and its successor, the DMG-DC system. We also introduce a real-world data mining application based on the DMG-DC: The Fleet Data Acquisition Miner (FDA-Miner) for analyzing the data generated by the Daimler fuel cell vehicle fleet. Finally, we present system evaluation results from the FDA-Miner and discuss related technologies.

II. MapReduce and Divide&Conquer

The tremendous amount of data generated in modern science and business applications require new strategies for storing and analyzing. As the amount of data increases, data can not be efficiently stored on a single storage server but has to be distributed to multiple machines. Google's MapReduce and its open-source implementations provide frameworks to mine these distributed data sets.

The name MapReduce refers to the map and reduce functions of functional programming languages. In the context of a MapReduce framework, all applications consist of a map and a reduce function [3]. The map function reads a key/value pair and produces a set of new key/value pairs. In an intermediate step all pairs are grouped by their key values. A key and its values are presented to the reduce function which produces a list of result values.
It can easily be shown, that these functions produce the same result when applied to the whole data set or to the parts of the data set.

MapReduce frameworks build an environment for executing these map and reduce functions on a cluster. Data is split up into small chunks and stored in a distributed file system comprised of multiple standard machines acting as storage *and* compute nodes [2]. A special manager node keeps track of all data chunks and their locations in the cluster. A master process manages the execution and minimizes data movement by executing the functions on the nodes containing the data to be mined. The master identifies the nodes to use for execution by asking the distributed file system manager for the location of the data chunks. If multiple copies of a chunk are available the master schedules the execution to the least used node.

Executing the functions on the nodes that contain the data is the key to the high performance and scalability of MapReduce frameworks. As not data, but algorithms are transferred, MapReduce frameworks are perfectly suited for Clouds because they do not require information about server location and network bandwidth as traditional systems need for data scheduling.

Restricting themselves to only two functions, MapReduce frameworks are easy to program and simple to set up. However, not all data-intensive applications can be decomposed into map and reduce functions. Especially the integration of existing data mining programs is sometimes impossible.

MapReduce can be viewed as a special form of the Divide&Conquer paradigm, where a problem is split into smaller sub problems that are easier to solve. This more general paradigm does not impose any restrictions on the functions or the number of processing steps. Applied to data-intensive applications, Divide&Conquer may be defined as follows: An arbitrary function f is executed in parallel on the selected subsets d of the data D:

$$f : D \rightarrow R, \quad f(d) = r_d, \ \forall d \in D$$

The results R may be processed by another function g,

$$g : R \times R \times ... \rightarrow S$$

generating the results S, which again may be processed by another function.
A series of such execution steps can be represented by a direct acyclic graph, where each node is a function and the vertices symbolize the data flow between the functions.

A data-intensive application based on this Divide&Conquer definition requires a distributed computer system providing:
(1) A distributed file system or a data registry to locate the subsets of the data. (2) A scheduler to execute the functions on the nodes containing the data subsets. (3) A workflow manager to coordinate the execution of each function in the direct acyclic graph.
Due to their specialized approach, the components of current MapReduce frameworks can not simply be reused to build a Divide&Conquer system, especially when such a system should be integrated in an environment comprised of heterogeneous, geographic distributed resources from multiple administrative domains. Therefore the flexible DataMiningGrid system was enhanced to natively support Divide&Conquer jobs.

III. Data Mining in the Grid: DataMiningGrid

In general, a grid-enabled data mining system should support the seamless and efficient sharing of data, data mining application programs, processing units and storage devices. As data mining is used by a wide variety of users and organizations such a system should not only address the technical issues but also pay attention to the unique constraints and requirements of data mining users and applications. In the DataMiningGrid project, use case scenarios from a wide range of application areas were analyzed to identify the key requirements of grid-based data mining that can be summarized as follows:
A grid-based data mining environment should offer benefits like increased performance, high scalability to serve more users and more demanding applications, possibilities for creation of novel data mining applications and improved exploitation of existing hardware and software resources.
Grid-enabling data mining applications should not require modification of their source code. The system should not

 ISBN: 978-1-61208-106-9

be restricted to specific data mining programs, tools, techniques, algorithms or application domains and should support various types of data sources, including database management systems (relational and XML) and data sets stored in flat files and directories.

To support the different user groups, intricate technological details of the grid should be hidden from domain-oriented users, but at the same time users with a deep knowledge of grid and data mining technology should be able to define, configure and parameterize details of the data mining application and the grid environment.

In order to address these requirements, the DataMiningGrid system was designed according to three principles: services-oriented architecture (SOA), standardization and open technology. The early adoption of two important distributed computing standards, the Open Grid Service Architecture (OGSA) and the Web Services Resource Framework (WSRF) were essential for succeeding projects, like the one presented in this article. The OGSA is a distributed interaction and computing architecture based on the concept of a grid computing service, assuring interoperability on heterogeneous systems so that different types of resources can communicate and share information. The WSRF refers to a collection of standards which endorse the SOA and proposes a standard way of associating grid resources with web services to build stateful web services required by the OGSA.

Following these principles, the DataMiningGrid project implemented various components based on existing open technology: Data management, security mechanisms, execution management and other services commonly needed in grid systems are provided by the Globus Toolkit 4 (GT 4) grid middleware.

Three higher-level components for data, information and execution management form the core of the DataMiningGrid system. The *data components* offer several data transformation and transportation capabilities to support typical data operations for data mining applications. The *Information Service* collects and manages all information about the data mining programs available in the system. The *Resource Broker* is responsible for matching available resources to job requests, global scheduling of the matched jobs and executing, managing and monitoring of jobs, including data stage in and out operations.

The main user interface is the Triana workflow environment. In combination with special data mining units, Triana enables users to build complex grid-based data mining applications.

The DataMiningGrid Application Description Schema (ADS) is the link between all the DataMiningGrid components. The ADS covers the complete life-cycle of a data mining program and is used for discovering, configuring and executing data mining programs.

Although the necessity to address data-intensive applications was recognized in the DataMiningGrid project, due to time constraints, the project focused more on compute-intensive applications. Hence, three functionalities needed for Divide&Conquer jobs are not available in the DataMiningGrid system:

1) The Resource Broker [6], like other grid resource brokers, is only able to schedule jobs to whole clusters. As a consequence, jobs can not be placed directly on certain machines inside a cluster, as required by Divide&Conquer jobs.
2) There is no specialized data registry that could be used for scheduling data-intensive applications.
3) There is no server-side workflow execution component to coordinate the steps of Divide&Conquer jobs.

The following section describes the changes made to the DataMiningGrid components and the new features of the DMG-DC system to support Divide&Conquer jobs in grid environments.

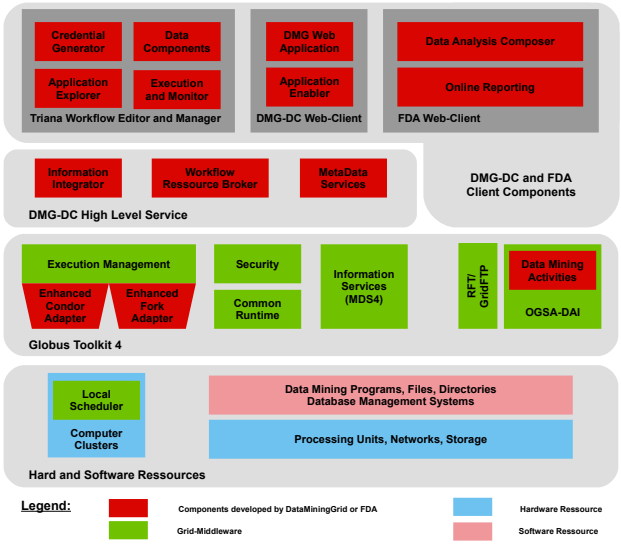


Figure 1. The DMG-DC architecture

IV. DMG-DC

The DMG-DC system is designed to support the different aspects of today's data analysis challenges. Based on the DataMiningGrid project, which already implemented many features needed for grid-based data mining, the DMG-DC development focused on the functionality to support extremely data-intensive applications. The flexible and extendable design of the DataMiningGrid system made it easy to integrate the missing functionality.

Consequently, the architecture of the DMG-DC, depicted in Figure 1, does not differ significantly from the DataMiningGrid [4], except for the new and redesigned components: The *Data Registry Service* (DRS) and the *Workflow Resource Broker* (WRB).

 ISBN: 978-1-61208-106-9

A. Data Registry Service

A data registry is a central component for executing Divide&Conquer jobs in a grid, as it provides the locations of all data sets to the Resource Broker. Without this information the Resource Broker would not be able to schedule jobs to the nodes containing the data to be mined. The developed distributed registry consists of a number of WSRF-compliant Data Registry Services that store user-defined metadata describing the data sets available in the grid. In contrast to the distributed file system of a MapReduce framework, the DRS only stores information about the data, leaving the actual storage to database management or file systems. Therefore, the Divide&Conquer mechanism can be applied to data stored in any storage system and is not limited to a specific distributed file system.

The DRS stores metadata in user-defined categories, which specify a list of logical and physical attributes describing the data. Logical attributes typically hold information about the content or creation process of the data set, whereas physical attributes include storage location, size or data format information. When a new data set is registered with a category, a logical and a physical object is created with unique object names. These objects contain the logical/physical attributes of that data set and are used to model replication: A logical object may reference any number of physical objects.

A single DRS may store the metadata information of all data sets in the grid. To improve reliability and performance several DRS may run on different sites in the grid. Multiple DRS automatically form a peer-to-peer network, forwarding client search requests and category information to the appropriate DRS.

A distinctive feature of the DRS, compared to other grid data registries like the Globus Toolkit Replica Location Service, is its advanced search mechanism enabling clients to search for data using multiple attributes within a single query.

B. Workflow Resource Broker

The new requirements arising from Divide&Conquer jobs led to the development of the DMG-DC Workflow Resource Broker. The WRB was designed not only to support Divide&Conquer jobs but also include all features of the DataMiningGrid Resource Broker [6]. The two main new features of the WRB are the workflow execution manager and the advanced job scheduler, able to place jobs on specific nodes inside a cluster.

As depicted in Figure 2, the WRB consist of 5 components communicating through well-defined interfaces:
Clients connect to the *workflow manager* to submit workflows, monitor and manage workflow execution. A workflow consists of one or more jobs, each described by an ADS instance, and dependencies between these jobs. The workflow manager is responsible for executing all jobs as

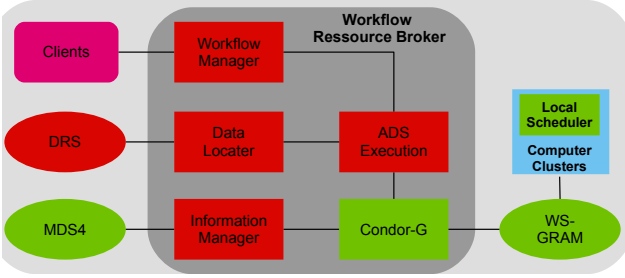


Figure 2. The components of the Workflow Resource Broker.

specified in the workflow. To start the execution of a single job, the workflow manager sends the corresponding ADS instance to the *ADS execution* component. The execution component analyzes the ADS instance and connects to the *data locator* to get the locations of all data sets specified in the ADS instance. The data locator acts as an interface to different data registries, although currently only DRS is supported. The execution component combines the data locations with the ADS definitions and generates a Condor-G job description. The job description contains all scheduling information necessary for placing Divide&Conquer jobs on nodes providing the selected data sets. *Condor-G* [7] is a powerful grid task broker providing advanced scheduling, execution and managing capabilities and an uniform interface to different grid execution management systems. A key feature of Condor-G is its ClassAd mechanism for describing and matching of jobs and compute resources. The flexible Condor-G ClassAd mechanism enables the ADS execution component to define a scheduling policy, resource and job parameters, so that Divide&Conquer jobs can be placed on a specific node in a cluster of the grid. The *information manager* collects all resource information necessary for this scheduling from the Globus Toolkit Monitoring and Discovery System (MDS4) and delivers it to Condor-G.
When receiving a job, Condor-G matches the job with all available resources in the grid and submits the job to the Globus Toolkit execution management service (WS-GRAM) providing the best match. Divide&Conquer jobs contain a special element that advises the WS-GRAM to start this job on the specified node(s), even if they are inside a cluster.

V. FDA-MINER

The presented DMG-DC system forms the basis of the Fleet Data Acquisition Miner (FDA-Miner) for analyzing the data generated by the Daimler fuel cell vehicle fleet. Daimler AG has been involved in fuel cell technology for more than 15 years and has released the largest fleet of zero emission fuel cell vehicles in the world with more than 100 vehicles [8]. The purpose of these operations is to test these vehicles in the hands of selected customers in everyday operations under varying climatic conditions, traffic conditions and driving styles in different locations

 ISBN: 978-1-61208-106-9

worldwide. In order to gain the most experience for future fuel cell vehicle development, a fleet data acquisition system has been developed which continuously records all relevant parameters of vehicle operation, such as the fuel cell voltage, current and temperatures. The enormous amount of world-wide distributed data produced by the fleet - over 4 million kilometers have been recorded - and the need for compute-intensive data analysis methods were the key drivers behind the development of the DMG-DC system [9].

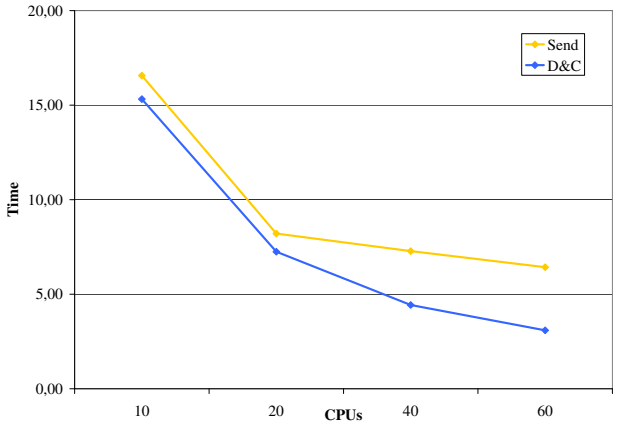
The FDA-Miner provides a user friendly web-based data analysis application for mining the fuel cell data. In addition to specialized visualization and reporting features, it offers a flexible front end to configure customized data mining tasks. The application uses the services of the DMG-DC to retrieve information about the available data and analysis programs. For each user defined task, the application creates the appropriate ADS instances and workflow definitions and submits a Divide&Conquer job to the WRB.

The FDA-Miner programming toolbox supports users implementing specialized Divide&Conquer data mining algorithms. The toolbox provides Perl and C modules to read the fuel cell data sets and templates for parallel data processing and combination steps.

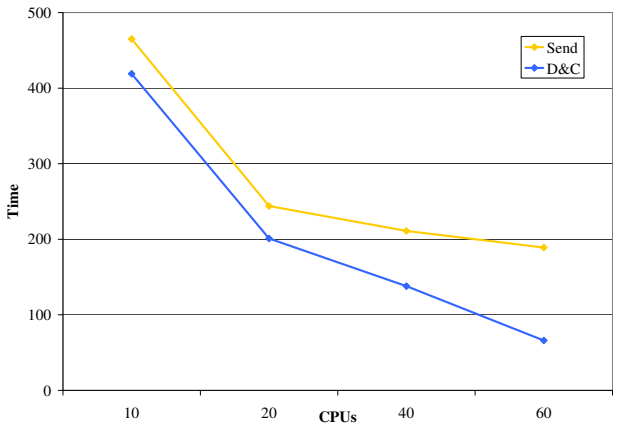
VI. Evaluation

The FDA-Miner has been already heavily used in production and successfully computed thousands of Divide&Conquer and other jobs. The following evaluation therefore focuses on the advantages of the Divide&Conquer functionality of the DMG-DC system compared to traditional grid systems, like the DataMiningGrid, with dedicated storage servers. The evaluation set-up consisted of 9 dual quad core machines with direct attached storage connected over a 1 GBit Ethernet network. To measure the performance of the DMG-DC Divide&Conquer functionality, a subset of the fuel cell data was randomly distributed over all 9 machines and each file was placed on at least two machines. Traditional grid systems were represented by a representative scenario with 1 storage server serving the data to 8 compute nodes.

A typical FDA-Miner data analysis job, filtering the data and computing various statistical properties, was executed on both set-ups. Figure 3 shows the overall time for performing this job while varying the number of CPUs and the size of the data set. The results demonstrate that the DMG-DC (blue curve), like MapReduce systems, scales well for common data analysis jobs. Traditional grid systems (orange curve) on the other hand have to send the data to the compute nodes first. Depending on the network bandwidth the transfer time may, for simple data filtering operations, even exceed the time for data processing. Scaling of these systems is also limited by the number of concurrent connections the storage server can handle without dropping network throughput. In the presented evaluation set-up the storage server can only deliver enough throughput to server about 20 CPUs and therefore does not scale well above that point.



(a) 11694 data sets



(b) 256530 data sets

Figure 3. Execution time of a job in Send and Divide&Conquer mode.

VII. Related work

Recently, various systems and approaches to grid-based data mining and MapReduce have been reported in the literature. Some of those, that are particularly relevant to the DMG-DC system, are briefly reviewed here.

GridMiner [10] is designed to support data mining and online-analytical processing in distributed computing environments. GridMiner implements a number of common data mining algorithms, some as parallel versions, and supports various text mining tasks. Two major differences between GridMiner and DMG-DC are the Divide&Conquer functionality and that the latter complies with the recent trend towards WSRF.

Knowledge Grid (K-Grid) [11] is a service-oriented system providing grid-based data mining tools and services. The K- grid system can be used for a wide range of

 ISBN: 978-1-61208-106-9

data mining and related tasks such as data management and knowledge representation. The system architecture is organized into a High-level K-Grid Services and a Core-level K-Grid Services layer, which are built on top of a Basic grid Services layer. K-Grid incorporates some interesting features for distributed data mining but no Divide&Conquer or similar functionality is available at the moment.

Hadoop [12] is the most well known open source implementation of Google's MapReduce paradigm. Hadoop's MapReduce framework is build on top of the Hadoop distributed file system (HDFS) containing all data to be mined. The map and reduce function are typically written in Java, but even executables can be integrated via a streaming mechanism. As MapReduce frameworks like Hadoop do not offer the functionality to execute compute-intensive applications (MPI, PVM) on the cluster, making them unsuitable for a general-purpose data mining system. Hadoop On Demand and Oracle Grid Engine try to overcome these limitations by running Hadoop on top of a cluster queuing system, thus adding another layer of complexity. Still, both reserve the nodes to use for MapReduce exclusively, making them unusable by other jobs. Hadoop and similar MapReduce frameworks simplify the development and deployment of data-intensive applications on local clusters and cloud resources . But, in contrast to the DMG-DC system, these frameworks are currently not suited for large-scale, heterogeneous environments with multiple independent virtual organizations.

VIII. Conclusion

In this article we introduced Divide&Conquer, a generalized MapReduce paradigm, for data-intensive applications. The developed DMG-DC system provides the functionality to run diverse data mining applications, including Divide&Conquer, in a worldwide, heterogeneous grid environment. As not data, but algorithms are transferred, Cloud resources can be used to scale the system on demand. The FDA-Miner, a real world data analysis application, uses the distinct features of the DMG-DC to efficiently mine the data of the Daimler fuel cell vehicle fleet. The FDA-Miner evaluation results highlight the advantages of the DMG-DC compared to traditional grid systems.

Future work may include the integration of more powerful data management systems like the Storage Resource Broker and a generalized version of the FDA-Miner programming toolbox.

References

[1] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "The kdd process for extracting useful knowledge from volumes of data," *Commun. ACM*, vol. 39, no. 11, pp. 27–34, 1996.

[2] S. Ghemawat, H. Gobioff, and S. T. Leung, "The google file system," *SIGOPS Oper. Syst. Rev.*, vol. 37, no. 5, pp. 29–43, 2003.

[3] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," 2004, pp. 137–150. [Online]. Available: http://www.usenix.org/events/osdi04/tech/dean.html

[4] V. Stankovski, M. Swain, V. Kravtsov, T. Niessen, D. Wegener, M. Röhm, J. Trnkoczy, M. May, J. Franke, A. Schuster, and W. Dubitzky, "Digging deep into the data mine with datamininggrid," *IEEE Internet Computing*, vol. 12, no. 6, pp. 69–76, 2008.

[5] K. Ranganathan and I. Foster, "Decoupling computation and data scheduling in distributed data-intensive applications," in *HPDC '02: Proceedings of the 11th IEEE International Symposium on High Performance Distributed Computing*. IEEE Computer Society, 2002, pp. 352–358.

[6] V. Kravtsov, T. Niessen, V. Stankovski, and A. Schuster, "Service-based resource brokering for grid-based data mining," in *Proceedings of The 2006 International Conference on Grid Computing and Applications*, Las-Vegas, USA, 2006.

[7] J. Frey, T. Tannenbaum, M. Livny, I. Foster, and S. Tuecke, "Condor-g: A computation management agent for multi-institutional grids," *Cluster Computing*, vol. 5, no. 3, pp. 237–246, July 2002.

[8] J. Friedrich, R. Schamm, C. Nitsche, J. Keller, B. Rehfus, T. Frisch, and M. Röhm, "Advanced on-/offboard diagnostics for a fuel cell vehicle fleet," in *Society of Automotive Engineers SAE World Congress 2008*, 2008.

[9] M. Röhm, J. Keller, and T. Hrycej, "Data mining fuel cell fleet data for stack degradation analysis," in *Fuel Cell Seminar, San Antonio*, 2007.

[10] B. Peter and W. Alexander, "Grid-aware approach to data statistics, data understanding and data preprocessing," *International Journal of High performance Computing and Networking*, vol. 1, no. 6, pp. 15–24, 2009.

[11] A. Congiusta, D. Talia, and P. Trunfio, "Distributed data mining services leveraging wsrf," *Future Generation Computer Systems*, vol. 23, no. 1, pp. 34–41, 2007.

[12] T. White, *Hadoop: The Definitive Guide*, 1st ed. O'Reilly Media, 2009.

 ISBN: 978-1-61208-106-9

Approach to Business-Policy based Job-Scheduling in HPC

Eugen Volk
High Performance Computing Center Stuttgart (HLRS)
Stuttgart, Germany
volk @ hlrs.de

***Abstract*—Job-Scheduling behavior of a High Performance Computing (HPC) provider is typically defined in the way that implicitly corresponds to its business policies. Represented mainly by a set of business rules or objectives, business policies form means to guide and control the business of HPC service provisioning. Because in HPC domain business policies exist mostly implicitly, administrators configure schedulers intuitively and subjectively. This makes it hard for business people to assess whether the actual scheduling behavior corresponds to current business policies, as there is no link defined between job-scheduling and business policies. The question, whether the scheduling behavior is configured correctly, cannot be answered without providing relationships between business-policies and job-scheduling strategies. Hence, more general question is: how much influence does business policy actually have on job scheduling? In this paper, we present an approach allowing investigating how business-policies relate to the job-scheduling in HPC domain.**

***Keywords*—*Business-policy, Job-Scheduling, Policy-based Management*.**

I. INTRODUCTION

Job-Scheduling behavior of a High Performance Computing (HPC) provider is typically defined in the way that implicitly corresponds to its business policies. Thereby, business-policies are a type of formal or informal behavioral guide prescribing behavior in a company, thus forming means to guide and control the business. Business-policies are usually formed by a set of business rules, business objectives, or in general, statements of control guides for delegated (to human or machine) decision making [1]. Business policies in the context of HPC affect several domains, such as security, accounting, SLAs, contracting, and others. They might have direct or indirect influences on job-scheduling. For instance, a business policy, such as "all jobs of premium customers have to be completed within 12 hours" has direct influence on scheduling, by determining the latest deadline of the job. Business-policies in HPC domain exist in most cases not explicitly, i.e., written by using domain specific language or natural language, but implicitly in the mind of the business people, whereas the configuration of job-scheduler is done by administrators.

The range of existing schedulers used for job scheduling in HPC varies from time-based scheduler like Cron [4] to advanced policy-based schedulers like Moab [3] or its open-source variant Maui, which support large array of scheduling policies. Scheduling policies define thereby behavior of the scheduler by, i.e., assigning priority to a job depending on job-size (number of CPUs or cores required), estimated job-duration, user's priority and other factors. However, schedulers have a big amount of parameters and different scheduling policies which need to be selected and adjusted in order to meet business policies in different situations.

A problem occurs when administrators are configuring schedulers. The configuration of schedulers is done in most cases intuitively and subjectively, because of implicit business policies, system administrators unaware of them, or in general, because of missing link or mapping between business policies and selection and configuration of scheduling policies. This makes it hard for business people to assess whether the actual scheduling behavior corresponds to current business policies, as there is no link between scheduling policies and business policies and it requires understanding of scheduling configuration parameters. The question, whether the scheduling behavior is configured correctly, can be answered by providing relationship between business-policies and job-scheduling policies. In this paper, we describe an approach allowing investigating how business-policies relate to job-scheduling in HPC domain and present intermediate results.

This paper is structured as follows. Section II presents related work in the area of job scheduling in HPC, policy-based management, and SLA based scheduling. Section III provides background information on job-scheduling in HPC. In Section IV we discuss the problem related to alignment of scheduling behavior with the business policies, showing the need for business-policy-based job-scheduling in HPC. Section V presents approach allowing investigating how business-policies relate to the job-scheduling in HPC and solve the problem described in previous section. Section VI provides intermediate analysis results achieved by applying proposed approach, identifying key-factors and relating them to business policies. Finally, the last section summarizes this paper and outlines work in progress.

II. RELATED WORK

In the target-area of "business-policy based job-scheduling in HPC" currently no work is known to the author. But, there is a weak relationship between "business policy-based job-scheduling" and "SLA (Service Level

 ISBN: 978-1-61208-106-9

Agreement) based job-scheduling", which is outlined below. However, there has been much work done in related areas: Job-scheduling in HPC, policy-based management, business-policies and SLA based job-scheduling.

In the area of job-scheduling in HPC, Iqbal, Gupta, and Fang [6] offer an overview about scheduling algorithms used for job-scheduling in HPC clusters. In [7], Casavant and Kuhl provide taxonomy of scheduling strategies in general-purpose distributed computing systems. In [8], Yeo and Buyya provide taxonomy of market-based resource management systems, citing over 79 references. In [9], Abawajy describes recent advances in efficient adaptive scheduling policies.

Many solutions in the area of policy-based management have been proposed. In [10], Boutaba and Aib provide history of policy-based management, referencing over 118 papers. In IBM's autonomic computing reference architecture [11], the authors drafted the principle on how policies on high level might be used to express business needs/objectives that govern IT infrastructure operations.

In the area of SLA-based job-scheduling many papers have been published. SLA is part of a service contract where the level of services or quality of services (QoS) is formally defined and agreed between service providers and customers. SLA contains usually rewards, for successful fulfillment of SLA, and penalties in case of SLA violations. SLAs are contracted in accordance with business-policies. In contrast to SLAs, business-policies prescribe, among others, kind of services and spectrum of QoS which can be offered principally to customers. Hence, SLAs can be considered as service level objectives contracted in accordance with the business-policies. On the other hand, business-policies are more prescriptive than SLAs, as SLA might be violated due to various reasons, but the behavior in a company must follow provider's business-policy. In [12][13][14][15], QoS and SLAs are used to find and allocate desired resources in quantity and quality, and determine priority and order of jobs for scheduling, among others, based on rewards and penalties declared in SLAs. In [13], authors describe how to derive IT management policies from SLAs, which in general follows autonomic computing approach (management by objectives).

In IBM's Whitepaper [16] authors provide most recent definitions of policies and rules in business area, relating them to IT. According to that definition, a "business policy" is a type of business directive that expresses the course of action that the business wants to have happen within a set of business conditions [16].

In conclusion, related work presented in this section outlined achievements needed to accomplish "business policy-based job-management in HPC" approach. As there is no work currently exist in the area of "business policy-based job-scheduling in HPC", we identified work in related areas. Approach in the area of policy-based management outlines hierarchical policy refinement process that transforms high-level policies into low-level policies. Similar methodology is used to achieve business-policy based job-management in HPC. Thereby, business policies represent high level policies that need to be transformed to low-level job-scheduling policies.

The bottom of the "business-policy based job-management in HPC" approach is formed by a work achieved in job-scheduling in HPC. This work presented scheduling algorithms and identified performance indicators needed to assess scheduling algorithms and policies. The top of the approach is formed by business policies, described and defined in IBMs' Whitepaper [16]. SLA-based job-management methodology applied policy refinement approach to SLAs and resource allocation policies in HPC domain. SLA-based job management methodology demonstrated how to allocate desired resources in quantity and quality for particular jobs, and determined, i.e., by sorting jobs according to rewards, penalties or deadlines stated in SLAs, the order of jobs. However, in contrast to SLAs, which are mostly related to fulfillment of the single job or set of jobs, business policies are type of business rules and directives used to control the whole process of HPC provisioning, involving other domains, such as: contracting, security, customer management, accounting, and resource management; they influence directly or indirectly job-scheduling behavior.

III. BACKGROUND

The cluster infrastructure of computing centers can be divided in two classes: high-throughput computing cluster and high performance computing cluster [6]. Nodes in high throughput computing clusters are usually connected by low-end interconnections. In contrast, more powerful nodes in high performance computing (HPC) cluster are interconnected by faster interconnection with higher bandwidth and lower latency. The application profile of high-throughput computing clusters includes loosely coupled parallel, distributed or embarrassingly parallel applications, requiring less communication and synchronization between nodes during the calculation. In contrast, the application profile of HPC clusters consists mainly of tightly coupled parallel applications, with high communication and synchronization requirements.

The computing nodes in cluster are managed by a resource management system (RMS), which is responsible for resource management, job queuing, job scheduling and job execution. Firstly, users who are willing to submit their applications or programs to resource management system need to express their applications as computational jobs, specifying requirements using, i.e., Job Submission Description Language (JSDL). Job specification contains usually number of nodes/CPUs/cores required, estimated maximum job-runtime, target architecture type (i.e., vector or scalar), specific I/O requirements (i.e., tools and files required for job execution) and other application or platform specific parameters. After expressing application as a job, user submits the job in batch to queue of the resource management system, where it waits in the queue with the jobs of other users, until it is scheduled and executed. Typically, a resource management system is comprised of a

 ISBN: 978-1-61208-106-9

resource manager and a job scheduler [6]. Most resource managers have an internal, built-in job scheduler, which can be substantiated by external scheduler with enhanced capabilities [6], i.e., with support for various scheduling policies like Maui [2]. Resource manager provides scheduler with information about job-queues, loads on compute nodes, and resource availability. Based on that information, scheduler decides on how and when to allocate resources for job execution. The decision of the scheduler follows scheduling policy that determines the order in which the competing users' jobs are executed. The order of jobs typically depends on job-size (amount of resources i.e., processors/cores required), estimated maximum job-runtime (indicated by user), resource access permission (established by administrator), resources available, and might depend additionally on QoS parameters (i.e., response time) expressed in contracts or SLAs. The assessment of scheduling behavior is typically done according to the following performance indicators [6][8][18]:

- **Wait time**: the time a job has to wait before the execution of the job starts
- **Response time**: how fast the user receives a first response from the system after the job is submitted
- **Turnaround time**: total time between when the job is submitted and when the job is completed. It includes wait time and execution time of the job.
- **Resource Utilization**: reflects the usage level of the cluster system
- **System Throughput**: number of jobs completed in a period of time

Typical performance criteria for users who expect minimal response time is the mean response time [6]. In contrast, administrators are typically trying to achieve maximum overall resource utilization, as that maximizes return on investment (ROI). Improving overall resource utilization and at the same time decreasing response time are two conflicting goals, as it requires that shortly submitted jobs with higher priority are executed as soon as possible, thus reducing the optimization space for efficient resource utilization.

IV. NEED FOR BUSINESS-POLICY-BASED JOB-SCHEDULING

Business policies are control statements that guide behavior in a company and control the business. Business policies are defined usually at an overall strategic level and can be related to specific areas. In HPC domain, these areas are: security, contracts and SLAs, resource management, accounting, and others. Business policies, which relate to security, contain statements governing the access to HPC resources, i.e., prescribing the process of obtaining permission to HPC resources, granting, restricting or refusing the access. Contract and SLA business policies contain statements, which i.e., describe the spectrum of performance and capacity capabilities of HPC provisioning offered principally. Resource management business policies contain statements influencing resource allocation and scheduling behavior on high level, i.e., by prescribing the preferences between users-groups.

As already mentioned, scheduling behavior is typically defined in the way that it implicitly adheres to business policies of HPC providers, while taking users' job requirements, available resources, existing SLAs, long term contracts and other factors into account. Advanced policy-based schedulers like Maui [2] have a big amount of parameters and different scheduling policies which need to be selected and adjusted in order to meet all business policies in different situations. As business policies exist mostly implicitly in the mind of people, or because administrators are not really aware of all of them, they configure schedulers intuitively and possibly subjectively. This makes it hard for business people to assess whether the actual scheduling behavior is correct and corresponds to current business policies, as there is no link between business policies and scheduling policies defined.

Additionally, there might be a fast switch required between different business policies. For instance, in profit oriented organizations, managers try to achieve maximum return on investment which often means that they only deliver various qualities of services to various users and groups [2] to increase system utilization. In contrast, nonprofit organizations, like national computing centers, have their focus on delivering various qualities of services to various (or certain) users and groups, even if this will cause a decreasing utilization. For instance, there could be situations where the cluster resources need to be exclusively reserved to a certain user, although no jobs on reserved resources might be executed during the reserved time-period. Some of the national computing centers have joint collaboration with scientific and industrial partners through common joint cooperation company. That means the scheduling behavior in clusters of such computing centers needs to be flexible enough to be adapted to various business needs, even at the same time.

Furthermore, there are cases where the usual job-scheduling behavior must be adapted to changing situations and requires evaluation of several business policies. For example, in case of fall-out of the cluster on which jobs of industrial users are executed, these could be shifted to another cluster, if allowed. The answer on the question whether the jobs of industrial users might be shifted, i.e., to research cluster, on which jobs of students or researchers are executed, depends thereby on evaluation of several business policies and facts. Research and educational clusters are typically financed by federal authority, whereas clusters used for industrial calculations are financed through common joint cooperation company. In case of the business policies, which prescribe that (1) industrial partners have higher importance than students or researchers, (2) only the owner (who has financed it) of the cluster may decide on its usage, and (3) current contract between federal land and HPC provider that allow usage of maximum 50 % of the cluster per month for industrial jobs, then the shifting of industrial jobs to the research cluster is allowed only if the 50 % limit is not exceeded.

 ISBN: 978-1-61208-106-9

As stated, there are many different business policies from different areas, which need to be considered when configuring schedulers. Furthermore, there might be a fast switch between different business policies required, and a fast adaptation of the scheduling behavior dependent on evaluation of several business policies from different domains. Because of implicit existence of business policies and missing link between business policies and scheduling policies, there is a risk of resulting incorrect scheduling behavior. In order to reduce the risk of miss-configuration we present an approach enabling to identify the link between business policies and scheduling policies, in next section.

V. APPROACH

An approach to handle problems described in previous section, induced by changing business objectives or altering situations, might follow IBM's autonomic computing reference architecture [11]. Autonomic computing is thereby defined "as a computing environment with the ability to manage itself and dynamically adapt to changes in accordance with business policies and objectives" [11]. Following this approach, there must be (1) business policies defined, capable to express business requirements influencing scheduling behavior on high level. Once, there are business policies defined, the next step (2) consist then of transforming these business policies with other sources (as SLA, Contracts, Accounting, etc.) influencing scheduling behavior into scheduling policies to configure advanced policy-based schedulers like Maui or Moab. In order to define business-policies explicitly, there must be HPC business policy specification language elaborated, capable to express business needs for various situations.

In order to address this problem, we will follow a bottom-up process:

The first step (1) consists of the analysis of existing scheduling policies in HPC in order to identify performance indicators (as described in section III) and key-factors like priority of user/customer (i.e., dependent on SLAs), accounting data, available resources, fairness etc. which influence scheduling behavior. In order to assess whether analyzed scheduling-policies make sense, there should be business-policies identified, which describe analyzed scheduling behavior on high level.

The next step (2) involves the analysis of existing business policies of particular high performance computing provider, in order to identify relationship to performance indicators and key-factors identified in the first step (1). The outcome of the second step will be a model, which explains relationships between specific business policies of particular HPC provider, performance-indicators, key-factors and scheduling policies. Especially the relationship between existing business policies and scheduling policies will provide an overview on how policy refinement process of transforming business policies to scheduling policies will principally looks like.

The third step (3) comprises the identification of HPC business policy schema, derived from the model developed in second step (2), capable to express HPC business policies influencing job-scheduling. Elements of the identified schema are used in the domain specific language, such as TEMPORA [19], to capture and model business policy specifications.

Finally, in order to evaluate results achieved in previously steps, the last step consists of the reference implementation, enabling mapping of reference business policies together with other key factors to scheduling policy configuration for advanced schedulers such as Moab [3] or Maui [2]. The implementation of the transformation rules, needed to translate business policies into scheduling policies, may be implemented using prolog engine, such as XSB [20].

The approach described in this section outlined steps of the transformation between business policies and scheduling policies. In order to illustrate this approach, we will present examples for steps (1) and (2) in the next section of the paper.

VI. FROM JOB-SCHEDULING-POLICIES TO BUSINESS-POLICIES

In this section, we present intermediate results achieved by applying the first two steps of the described methodology. Firstly, we analyze briefly job-scheduling in HPC, presenting common scheduling algorithms and policies in subsections A and B. In Subsection C, we identify key-factors, influencing scheduling behavior from different point of views. In Subsection D, we investigate relationships between key-factors and business policies, illustrating mapping between business policies and job-scheduling policies in few examples.

A. Analysis on Job-scheduling in HPC

Job-Scheduling algorithms or policies can be divided in two classes: time-sharing and space-sharing [6]. Time sharing algorithms divide time on a processor into several slots, each time-slot is assigned to unique job then. In contrast, space-sharing algorithms assign requested resources to unique job, until job is completed. In all HPC clusters is space-sharing approach used mostly, as time-sharing approach increases synchronization overhead between nodes of the same job.

The simple space-sharing algorithms are [6] first in first out (FIFO), first come first serve (FCFS), shortest time job first (STJF), longest time job first (LTJF), largest job first (LJF) etc. FIFO and FCFS execute jobs in the order in which they enter the queue. In case, there are not sufficient resources available to start a job, FCFS waits, until required resources are available. STJF periodically sorts the incoming jobs in the queue assigning jobs with the shortest estimated running time for the execution. LTJF sorts periodically the incoming jobs and assigns jobs with the longest estimated running time for the execution. LJF sorts incoming jobs periodically assigning jobs with the highest number of nodes/cores required for the execution. Additionally, there might be a priority to each job assigned, with the aim to reduce response time, as job with higher priority are executed prior lower priority jobs.

The simple scheduling algorithms might be enhanced by combining them with the use of advanced reservation and backfill techniques. Advanced reservation algorithms use

 ISBN: 978-1-61208-106-9

estimated job-runtime to make reservation on resources for particular jobs and create time-schedule for certain time period. The problem thereby is that schedule is based on estimated job-runtime, which is in most cases much longer than the real one. That means the schedule needs to be adapted as soon as jobs are completed earlier than expected. The backfill strategy improves basic strategies by combining them with additional iteration to fill out the gaps. Given schedule on high priority jobs i.e., by applying LTJF strategy, the scheduler use in second iteration lower priority jobs to fill out the gaps (free time slots on unused resources) between higher priority jobs.

B. Policy-based Job-Scheduling

In order to enable administrator to control and adapt scheduling behavior (when, where and how resources are allocated to jobs) more fine granular and more quickly to different situations and needs, there exist policy based job-schedulers like Maui [2] and Moab [3]. Policies include in particular for Maui [2]: job prioritization, allocation policies, fairness policies, fairshare configuration policies, and scheduling policies. These are explained in detail in [2].

C. Identifying Key-Factors

Analyzing the scheduling algorithms and policies leads to identification of key-factors, characterizing (and determining) scheduling behavior from different point of view: customer, provider, and administrator of the cluster.

Customer-centric key-factors are: turnaround-time, response-time, meeting deadlines, and exclusive resource reservation.

Provider centric key-factors need to differentiate between maximum return on investment (ROI) and customer satisfaction. Hence provider centric key-factors are: resource-utilization (in case of max ROI) or high job-throughput, and customer satisfaction (trying to satisfy customer centric key-factors).

Administrator centric key-factors are: achieving provider's goal by configuring job-queue and identifying right scheduling policies based on provider's preferences.

In addition, there are independent key-factors, which form the scheduling situation: available resources (quantity and quality of resources), job complexity (quantity and quality of resources required for job-execution and job-execution-time).

These key-factors are in the next step related to business policies, explained in next section.

D. Key-factors and Business-Policies

As outlined in the previous section, there is a need for business policy specification capable to express business needs in order to adapt scheduling behaviour to new situations, without need to understand scheduling configuration parameters in detail. Considering all those key-factors identified in previous section from the business point of view, they might be divided in two categories *decision making* and *optimization of scheduling* behavior. Decision making affects principal question on how the customers (including what kind of customers/users) are supplied with services and what kind (QoS) of services can be offered/delivered to customers. Scheduling optimization criteria determine the focus of scheduling optimization. A possible taxonomy of business policies is outlined below.

Decision making includes prioritization (between users or their jobs), reservation of resources (to certain user), meeting deadlines, fairness, preemption.

Optimization of scheduling behavior comprises: optimization criteria (ROI / resource utilization, customer satisfaction, energy efficiency, etc.), prioritization between criteria and expression to what degree criteria might be not fulfilled.

Typical Business Policies might looks like and mapped to scheduling policies as follow:

Industrial users are preferred against the scientific users or students. This will be mapped to scheduling policies as follow: jobs submitted by industrial user-group have higher priority than the jobs of scientific or student user-group.

Jobs of GOLD customers must have response time of X hours. Jobs for all users of user-group GOLD must be started latest after X hours after the job submission. In order to fulfill such kind of business policy, there might be a dedicated job-queue for GOLD customers created. The amount of resources granted to GOLD job-queue depends thereby typically on mean job-size and amount of customers/users of type GOLD. In critical case there might additional resources allocated from other job-queues.

VII. SUMARY AND FUTURE WORK

In this paper, we outlined why business policy-based jobs-scheduling is needed, and presented an approach allowing to investigate how business policies relate to job-scheduling in HPC domain. The proposed bottom-up process explains identification of relationships between scheduling policies and business policies in several steps, including scheduling-performance-indicators, and key-factors. The process includes also elaboration of business policy language, capable to express business policies in HPC. The general aim of the proposed approach is to realize hierarchical policy refinement, allowing transformation of business policies together with other constrains into selection and configuration of parameters and policies needed to configure policy based schedulers. Intermediate results outlined in Section VI showed how identified key-factors, characterizing and determining scheduling behavior, might relate to business policies.

The approach and results presented in this paper are part of currently ongoing PhD work. The scope of ongoing and future work comprises all steps stated in Section V.

ACKNOWLEDGMENT

The results presented in this paper are partially funded by Federal Ministry of Education and Research (BMBF) through the TIMaCS project [17]. TIMaCS deals with the challenges in the administrative domain upcoming due to the increasing complexity of computing systems especially of

 ISBN: 978-1-61208-106-9

resources with performance of several petaflops [17]. The business policy-based management approach presented in this paper is part of the extended TIMaCS vision to use business policies for differentiated and goal oriented system management, while ensuring correct system management (in accordance with the business policies of the provider).

REFERENCES

[1] W. T. Greenwood, "Business Policy-Case Method Forum: A Rejoinder", in *The Academy of Management Journal*, Vol. 10, No. 2 (Jun., 1967), pp. 199-204

[2] Maui Scheduler Administrator's Guide, version 3.2 from http://www.clusterresources.com/products/maui/docs, access on 08.09.2010

[3] Moab Workload Manager user-guide, http://www.clusterresources.com/products/mwm/docs/moabusers.shtml, access on 03.09.2010

[4] Cron Wikipedia description, from http://en.wikipedia.org/wiki/Cron, access on 08.09.2010

[5] IBM, "Policies and Rules improving business agility", IBM website, http://www.ibm.com/developerworks/webservices/library/ws-policyandrules/index.html, access on 16.06.2010

[6] S. Iqbal, R. Gupta, and Y. Fang, *Planning Considerations for Job Scheduling in HPC Clusters*. Dell PowerSolutions, Feb 2005

[7] T. L. Casavant, G. J. Kuhl, "A taxonomy of scheduling in general-purpose distributed computing systems". *IEEE Transactions on Software Engineering* 1988; 14(2):141–154.

[8] C. S. Yeo, R. Buyya, "A taxonomy of market-based resource management systems for utility-driven cluster computing." in *Software-practice and experiences* 2006; 36:1381–1419, Published online 8 June 2006 in Wiley InterScience

[9] J. H. Abawajy, "An efficient adaptive scheduling policy for high-performance computing", in *Future Generation Computer Systems*, Volume 25, Issue 3, March 2009, pp. 364-370.

[10] R. Boutaba, I. Aib, "Policy-based Management: A Historical Perspective", *Journal of Network and Systems Management*, pp. 447-480, Springer, 2007

[11] IBM, "An architectural blueprint for autonomic computing.", *IBM Whitepaper*, June 2006, http://www-01.ibm.com/software/tivoli/autonomic/pdfs/AC_Blueprint_White_Paper_4th.pdf

[12] R. Sakellarioiu, V. Yarmolenko, "Job Scheduling on the Grid: Towards SLA-Based Scheduling." in *High Performance Computing and Grids in Action*, pp. 207–222. IOS, 2008.

[13] V. Yarmolenko, R. Sakellariou, "An Evaluation of Heuristics for SLA Based Parallel Job Scheduling." *3rd High Performance Grid Computing Workshop* (in conjunction with IPDPS 2006), 2006.

[14] J. Sherwani, N. Ali, N. Lotia, Z. Hayat, R. Buyya, "Libra: Economy-Driven Job Scheduling System for Clusters.", in *Software: Practice and Experience* 2004; 34(6):573–590.

[15] L. Tang, Z. Yang, Z. Yu, Y. Wang, "A Quality-Driven Algorithm for Resource Scheduling Based on Market Model on Grid.", 2007 International Conference on Parallel Processing Workshops (ICPPW 2007)

[16] M. Hondo, J. Boyer, A. Ritchie, "Policies and Rules – Improving business agility: Part 1: Support for business agility", IBM Whitepaper, 16. March 2010

[17] TIMaCS - Tools for Intelligent Management for Very Large Computing Systems, web site: www.timacs.de

[18] Scheduling Wikipedia description, from http://en.wikipedia.org/wiki/Scheduling_(computing), access on 08.09.2010.

[19] P. M. Brien, M. Niezette, D. Pantazis, A. H. Seltveit, U. Sundin, B. Theodoulidis, G. Tziallas, and R. Wohed, „A Rule Language to Capture and Model Business Policy Specificatins., in *Proceedings of the third international conference on Advanced information systems engineering*, 1991, pp. 307 – 318.

[20] XSB sourceforge project web site, http://xsb.sourceforge.net/, access on 08.09.2010

 ISBN: 978-1-61208-106-9

An Efficient Job Scheduling Technique in Trusted Clusters for Load Balancing

Shakti Mishra, Dharmender Singh.Kushwaha, Arun Kumar Misra
CSED, MNNIT Allahabad India
{shaktimishra,dsk,akm}@mnnit.ac.in

Abstract—**Although there has been tremendous increase in PC power and most of it is not fully harnessed, yet certain computation intensive application tend to migrate their process with the aim of reducing the response time. Cluster computing is the area that aims just at this. Clustering provides means to improve availability of services, sharing computational workload and performing computation intensive application. However, these benefits can only be achieved if the computing power of cluster is used efficiently and allocated fairly among all the available nodes. As is the case with our desktops, it is usually seen that clusters also suffer from underutilization. A number of approaches proposed in the past share only idle CPU cycles and not use the resources of systems when the machine has its own local processes to execute. We propose a priority-based scheduling approach for run queue and multilevel feedback queue scheduling approach for migrated tasks that doesn't degrade the performance of local jobs too. Simulation and experimental results have been able to show that priority-based run queue management and multilevel feedback queue scheduling for migrated tasks can increase overall throughput by about 28-33 percent.**

Keywords- Cluster;Load Balancing; Scheduling; Priority; Multilevel feedback Queue.

I. INTRODUCTION & SURVEY OF RELATED WORK

Clustering provides a better alternative to High Performance Computing (HPC) since the cost of highly available machines such as idle workstations and personal computers is significantly low in comparison to traditional supercomputers [1]. It has potential to improve availability of services, sharing computational workload and performing computation intensive application through efficient resource usage. The computational requirement of various applications can be met using cluster technology in an effective manner. However, these benefits can only be achieved if the computing power of clusters used efficiently and allocated fairly among all the available nodes.

Most of the machines do not use their full CPU capacity at any point of time. So, the fundamental policy of each machine in computer supported co-operative working (CSCW) is to share idle CPU cycles of these machines with any remote process which is demanding for CPU while not deteriorating the performance of original machine. But at the same time, the proposal has a constraint that the resources of systems can't be shared when the machine has its own local processes to execute. This becomes bottleneck when real time processes arrives on a machine. These remote real time processes begin to starve since they can't be scheduled on machine until unless CPU becomes idle. Thus, proper mixing of local and remote processes ensures no starvation policy for both.

Scheduling and interleaving of tasks in an optimal manner is mandatory for utilizing full capability of computing nodes with reduced completion time. The goal of the scheduling is to exploit the true potential of the system.

Cluster based distributed systems resolves complex problems by partitioning the task into sub tasks and then scheduling them in such a way so that each machine is assigned equal work and thus, balancing the load across the cluster with reduced waiting and response time, while ensuring little migration overhead.

The computing environment of nodes depends upon the cluster usage pattern that is broadly classified as Network of Workstations (NOW) and PMMPP (poor man's Massively Parallel Processors) (Table 1). NOW mode of cluster usage pattern is based on using idle cycles of personal computers or workstations and this implies an infinitely higher priority for workstations owner processes over remote processes (migrated processes from other workstations) [8]. MPP mode involves dedicated cluster for execution of high performance application.

TABLE I. CLUSTER USAGE PATTERN

Mode	Type Of Workload	Workstations Usage	Major Projects
Network of Workstations (NOW)	Regular Workload HPC Workload	Idle Cycles	CONDOR, MOSIX
MPP (Massively Parallel Processors)	HPC Workload	Dedicated cluster for HPC application	Beowulf, RWC PC

Typically, the jobs arrived on various workstations as a result of load balancing, contains different processes that may be dependent or independent from each other.

Depending upon these different type of processes, scheduling decisions may vary. Basically, scheduling choices are based upon two facts; (a) Number of processors available, (b) Process type (typically involves communication and synchronization pattern of processes). Number of processors available is further categorized as bounded number of processors or unbounded number of processors [3].

The author in [7] discusses about minimization of migration cost and defines a strategy as to which parts of the program should migrate. Many of the researchers [10] have tried to resolve issues like longer freeze time that may be due to unavailability of competing resources but their approach resolves pre-fetching of memory pages for process migration.

 ISBN: 978-1-61208-106-9

Although various approaches for scheduling tasks in clusters have been proposed and implemented by previous researchers [6, 9]. We find that each scheduling approach has its own assumption; however, for a load balancing system, we propose a combined approach, priority-based run queue management and multilevel feedback queue (MLFQ) scheduling approach for migrated tasks. Authors in [11, 12] claim that Multilevel Feedback Queue (MLFQ) scheduling proves viable for general purpose systems, however in this scheme CPU intensive processes suffers from starvation. MLFQ scheduling is chosen because of the several advantages as following:

- MLFQ uses priorities to decide which job should run at a given time: a job with higher priority (i.e., a job on a higher queue) is the one that will run.
- MLFQ uses the history of the job to predict its future behavior.
- MLFQ scheduling uses priority boost technique to raise the priority of processes to ensure that no process starves due to lower priority.

The proposed scheduling approach tries to resolve following issues:

- To prevent frequent migration of process due to unavailability of nodes by ensuring proper mixing of local and remote processes in run queue.
- Identification of critical processes by assigning highest priority and scheduling these processes immediately on one of the available processors.
- No starvation policy for any process while considering the priority and criticality of process.
- Scheduling local and remote jobs in run queue with same priority in round robin fashion so that neither of these processes may starve.
- Boosting priority of computation intensive remote processes in MLFQ to reduce remigration and congestion across the network.

The rest of the paper is organized as follows. Section 2 describes system model. In Section 3, process scheduling algorithms investigated in this paper are discussed. The performance analysis of algorithms is carried out in Section 4, followed by conclusion.

II. SYSTEM MODEL

A. Base Model

In our previous work [14], we have proposed that the cluster contains group of trusted nodes. A common node between two clusters is selected as Process Migration Server which we will now referred as Process Management Server (PMS). In case, if no node overlaps in two cluster, then a least loaded node would be referred as PMS as discussed in [13] . In order to reduce the overhead of polling and broadcasting periodically, each node of the cluster sends its load status information to PMS, when it changes. As node sends their load statistics, PMS updates its Node_Status_Table (NST) (Fig.1). PMS has several daemon processes which handle following functions:

1. Collecting load statistics from all other nodes and maintaining and updating NST.
2. Registering each node via registration module and maintaining trust among all nodes [4, 5].
3. Group multicasting the list of least loaded nodes to overloaded nodes.

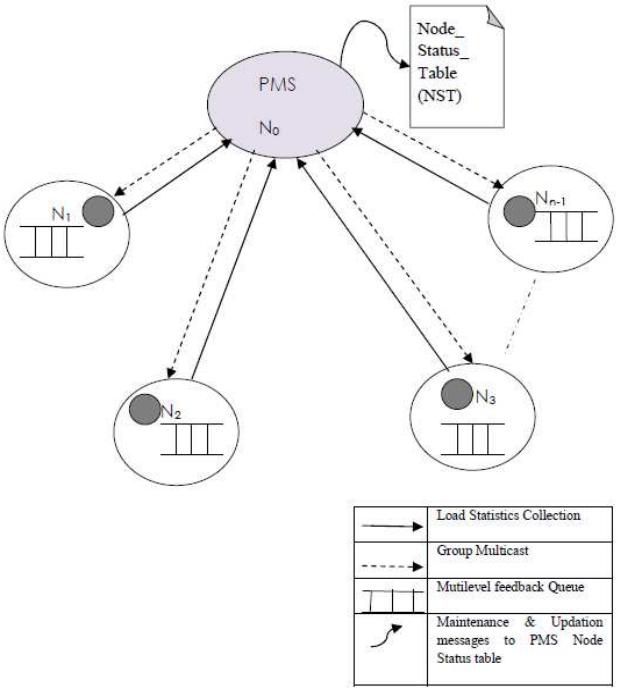


Figure 1. Base Model of JMM with MLFQ

This has been illustrated in Fig. 1 in which trusted nodes send their status updates information to PMS. A local process scheduler computes various parameters on each node like CPU utilization, resource availability including input / output resources, network bandwidth and memory usage including the number of processes in the process queue. CPU utilization refers to the CPU contribution to the functioning of a node. It is considered to be high if less number of CPU cycles is wasted. Each node also has an updated status of resources available in the system.

B. Local Scheduler

A local scheduler is responsible for maintaining multilevel queues on each node as shown in Fig. 2. The objective is to locate a process in the highest priority queue and assign the CPU to it. It is invoked, directly or in a lazy way, by several kernel routines. The scheduler keeps track of what processes are doing and adjusts their priorities periodically. When a resource request or migration request arrives from some node, the scheduler keeps these processes in the highest priority queue and adjusts their priority repeatedly and also checks whether the resource needed by the process is available; if not, it yields the CPU to some other process by invoking scheduler [2].

 ISBN: 978-1-61208-106-9

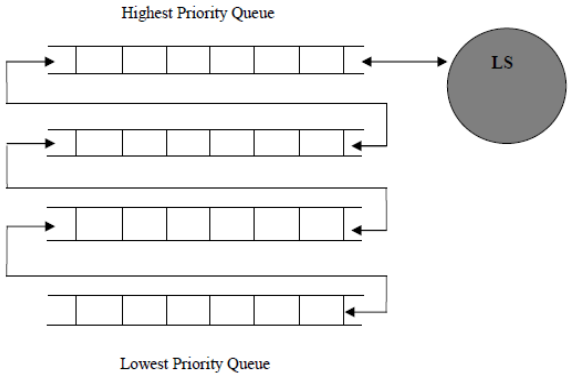


Figure 2. Design of Local Scheduler

The local scheduler must have information about available and occupied resources. There may be some jobs in which node status is underloaded while the node is suffering from memory unavailability. In this case, the process would migrate due to insufficient memory.

III. PROCESS SCHEDULING STRATEGIES

A. Node Selection Strategy

- *Least Busy CPU First (LBCF):* On the basis of information provided by PMS, an overloaded node may choose least busy node for migration.
- *Neighbor CPU First (NCF):* PMS maintains a closest vector node for each of the node in the state table. The criteria may be chosen as minimum number of hops from overloaded node to an idle node.
- *Random Selection:* An overloaded node can randomly choose any idle node for process migration. The selection criteria may be the resource availability, memory availability, network bandwidth, compatibility among system and many other factors. This random selection is based upon greedy approach.

B. Scheduling Strategy

Our proposed scheduling strategy is partitioned into two sections. The first section deals with dynamic run queue management by appropriately loading processes to main memory from process pool while second one deals with run queue CPU allocation to processes. CPU scheduler is a critical piece of the operating system software that manages the CPU resource allocation to tasks. It typically strives for maximizing system throughput, minimizing response time, and ensuring fairness among the running tasks in the system [16]. Our proposed scheduling strategy is based on the priority and criticality of the processes. The term critical process refers to a process whose execution should not be delayed or the process which has strict time constraints. Such processes carry highest priority. We propose that if the process is critical, its execution should only be suspended if the node that receives this process is executing its own critical local/remote process.

1) Priority-based Run Queue Management

Based on the above discussion, we have four types of processes. The convention for these processes is listed in Table 2.

TABLE II. PROCESS CONVENTION

S. No.	Process Type	Convention
1	Local Process	Li
2	Remote Process	Ri
3	Local Critical Process	C Li
4	Remote Critical Process	CRi

The process scheduler selects the job from the pool using one of the following cases:

[Case A] When an idle node receives a migration request of (local or remote) critical process, it is immediately chosen for execution.

[Case B] When a remote process (R_i) gets migrated on idle node, it executes on remote machine until unless a critical local process arrives.

If a critical local process arrives on the same machine while execution of a remote process, then it preempts the remote process to waiting queue. However, if in the mean time, local process waits for an I/O resource, the remote process dequeues from waiting queue and get chance to execute. Linger-Longer approach [8] provides this facility for fine-grained idle periods to run foreign jobs with very low priority.

[Case C] An underloaded node has critical local process to execute and simultaneously it gets a migration request of critical remote process. Then, it simply rejects the request.

[Case D] A local process arrives on an underutilized node currently executing an remote process, then both processes start their execution in round robin fashion.

We consider the case where a remote job is being executed on an idle node and simultaneously local process arrives. Condor [15] uses pre-emption technique to resolve this problem while the approach does not consider the case of starvation of remote process if the frequency of local process is too high. Here, we follow the round robin scheduling. This approach scores over previous systems that support collaborative working while utilizing resources and CPU efficiently with no starvation. The scenarios discussed in CASE A and CASE B can be described by the data shown Table 3 which shows different processes with equal priority and their arrival time with their execution time.

TABLE III. SCHEDULING DATA FOR CASE[A]

Process	Arrival Time	Execution Time
L1	0	3
L2	1	1
L3	4	4
CR1	5	2
R1	6	1

The scheduling policy for above shown processes can be described as Fig. 3. Initially, the local process executes as per round-robin fashion. However, at time unit 5 when Critical remote process (CR1) arrives, scheduler preempts the CPU from local process L3 and CR1 starts its execution.

 ISBN: 978-1-61208-106-9

As CR1 finishes its execution, local process L3 and remote process R1 continue their execution in the round robin order.

Let us consider another case with different processes and their characteristics as depicted in the table (Table 4) given below. Here the initial execution of local and remote processes is same as Fig. 4 until remote critical process R1 arrives. As CL1 arrives, all local and remote processes are put to waiting queue and CL1 starts its execution. Now, if at the same time critical remote process (CR1) arrives, the request is discarded by scheduler.

TABLE IV. SCHEDULING DATA FOR CASE [B] & CASE [C]

Process	Arrival Time	Execution Time
L1	0	3
L2	1	1
L3	4	4
R1	5	2
CL1	6	3
CR1	7	1

Now, consider Table 5. with different type of processes and their priority with their arrival and execution time,

TABLE V. SCHEDULING DATA FOR CASE[D]

Process	Arrival Time	Execution Time	Priority
L1	0	3	1
L2	1	1	2
L3	4	4	2
R1	5	2	3
CL1	6	3	0

The scheduling mechanism (Fig. 5) depends upon the priority of processes. At time interval 0, L1 starts its execution. At time interval 1, L2 arrives. Since, the priority of L1 is greater than from L2, L1 continues its execution and L2 waits. At time interval 3, L2 starts its execution and then L3. Remote process R1 arrives at 5 unit of time, scheduler compares the priority of R1 and L3, since R1 carries lower priority, it is put in to the waiting queue. At time interval 6, critical local process (CL1) arrives, scheduler preempts the CPU from local process L3 and CL1 starts its execution. As CL1 finishes its execution, local process L3 finishes first and then and remote process R1 continues its execution.

2) Multilevel Feedback Queue Approach for Migrated Tasks

A remote job aware multilevel feedback queue based scheduling for migrated tasks is shown in Fig. 6.

A multilevel feedback queue consisting of five queues; each assigned a different time quantum, the CPU switching time and the priority levels are given in the Table 6.

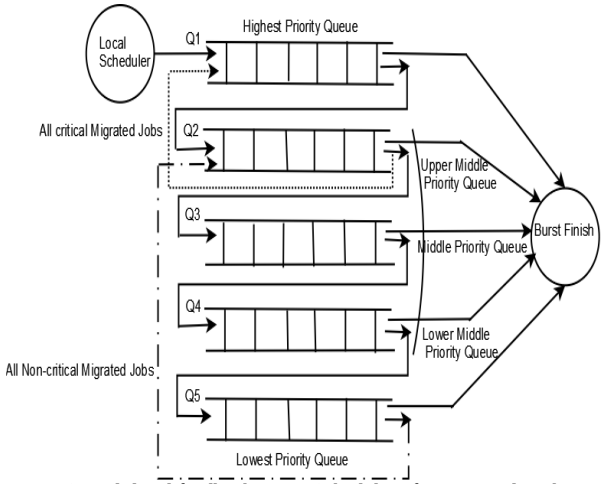


Figure 6. Multilevel feedback queue Scheduling for Migrated Tasks

TABLE VI. MLFQ FOR MIGRATED TASK PARAMETES

Queue	CPU switching time (ms)	Priority level
Q1	16	Highest
Q2	32	Upper Middle
Q3	64	Middle
Q4	128	Lower Middle
Q5	256	Lowest

IV. PERFORMANCE EVALUATION

We computed the average waiting time and mean response time for local and remote processes. Migration overhead is considered negligible in this analysis to simplify the model and simulations.

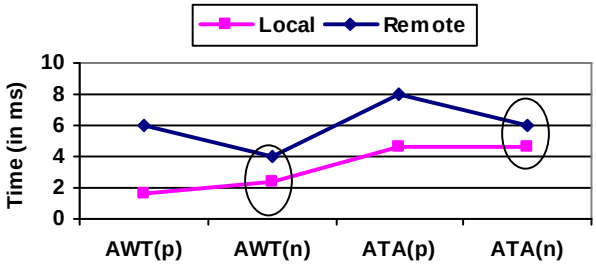


Figure 7. Average Waiting & Turnaround Time Comparison of previous approach with proposed new approach for data given in Table3.

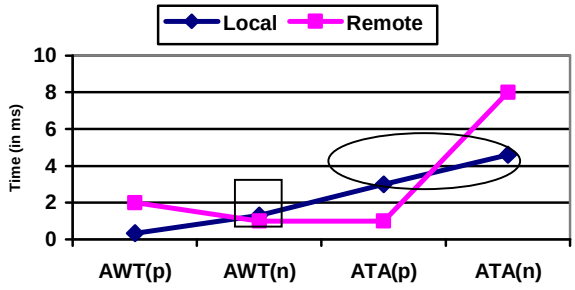


Figure 8. Average Waiting & Turnaround Time Comparison of previous approach with proposed new approach for data given in Table5.

Fig. 7 shows the comparison of average waiting & turnaround time of proposed new approach (AWT (n) & (ATA(n)) with previous approaches (AWT(p) & ATA(p)). We observe from the encircled values in Fig. 7 that proposed approach is able to reduce the average waiting and turnaround time of local and remote processes by 28-33%.

The simulation analysis proves that for data set given in Table 5, previously proposed approaches didn't accept any remote process when the machine had its own local processes to be executed. With our approach, the system is able to accommodate remote process along with local process with marginal increment in the waiting & turnaround time of local processes.

The values enclosed in square area in Fig. 8 shows that using our proposed approach, the average waiting time of remote processes is decreased in comparison to others, and significantly lowered as compared to local processes. From the values encircled in the same, we also observe the reduced turn around time of remote process with allowable increment in the turnaround time of local processes.

V. CONCLUSION

This paper presents an optimized scheduling approach for trusted cluster environment that resolves important issues of remote process starvation in case of local processes arrival, frequent migration of remote processes and selection criteria of idle node.

The experimental results establish that priority-based scheduling increases overall throughput by about 28-33 percent. In some cases, we also find that the proposed approach is able to accommodate more number of remote processes than the previous approaches with marginal increment in waiting and turnaround time of local processes. This in turn shall allow more users going the cluster computing way without the concern of degraded system performance and further investments in scalability and redundancy.

REFERENCES

[1] R. Buyya, "High Performance Cluster Computing: Architecture and Systems" Vol. 1. Pearson Education, pp. 61-68.

[2] O'Reilly, "Understanding the Linux Kernel", O'Reilly Media, 2002.

[3] S. Pasham and W. Lin, "Efficient task scheduling with duplication of bounded number of processors", 11th International Conference on Parallel and Distributed Systems (ICPADS'05)vol. 1, pp. 543-549. 2005

[4] Shakti Mishra, D.S.Kushwaha, and A.K.Misra," A Cooperative Trust Management Framework for Load Balancing in Cluster Based Distributed Systems", In IEEE proceedings of International Conference on Recent Trends in Information, Telecommunication and Computing, ITC 2010. pp. 121-125. March 2010

[5] Shakti Mishra, D.S.Kushwaha, and A.K.Misra," A Novel Approach for Building a Dynamically Reconfigurable Trustworthy Systems", Information Processing and Management: Proc. of International Conference on Recent Trends in Business Administration and Information Processing, BAIP 2010, CCIS 70, Springer-Verlag, Berlin Heidelberg, pp. 258-262. March 2010.

[6] H. Rajaei, "Simulation of Job scheduling for small scale clusters", Proc. of Winter Simulation Conference, pp. 1195-1201. 2006.

[7] H. Karatza, "A comparison of load sharing and job scheduling in Network of workstations", I. J. of Simulation Vol. 4 (3-4), pp. 4-11. 2003.

[8] K. D. Ryu and J. K. Hollingsworth, "Exploiting Fine-Grained Idle Periods in Network of Workstations", IEEE Transactions on Parallel and Distributed Systems, Vol.11, No.7,, pp. 683-698. July 2000.

[9] T. Akgun, "BAG Distributed Real-Time Operating System and Task Migration", Turkish Journal Electrical Engineering, Vol. 9, NO. 2, pp.123-136. 2001.

[10] Ho, R.S.C., Cho-Li Wang, and Lau, F.C. "Lightweight Process Migration and Memory Prefetching in Open MOSIX", IEEE International Symposium on Parallel and Distributed Processing IPDPS, pp. 1-12. 2008.

[11] Arpaci Dusseau, "Scheduling Multilevel feedback Queue", Operating System, Ch. 7, pp. 1-8.

[12] K. Hoganson, "Reducing MLFQ Starvation with Feedback and Exponential Averaging", Southeastern Conference on Consortium for Computer Science in Colleges (CCSC), Vol.25 Issue 2. pp. 196-202. Dec' 2009.

[13] Shakti Mishra, D.S.Kushwaha, and A.K.Misra, "Hybrid Load Balancing in Auto-configurable Trusted Clusters", Journal of Computer Science and Engineering, Vol. 2 (1), pp. 16-25. July 2010.

[14] Shakti Mishra, D.S.Kushwaha, and A.K.Misra," Jingle- Mingle: A Hybrid Reliable Load Balancing Approach for a Trusted Distributed Environment". 5th IEEE International Joint Conference on INC, IMS & IDC, NCM 2009, pp. 117-122. Aug' 2009.

[15] M. J. Litzkow, M.Livny, M. W. Mutka, "Condor-A Hunter of Idle Workstations", 8th IEEE International Conference of Dstributed Computing and Systems, pp. 104-111. 1988.

[16] Siddha S., Pallipadi V., Mallick A., "Process Scheduling Challenges in Multicore Processors", Intel Technology Journal, Vol. 11 , Issue 04, pp. 361-369. 2007

 ISBN: 978-1-61208-106-9

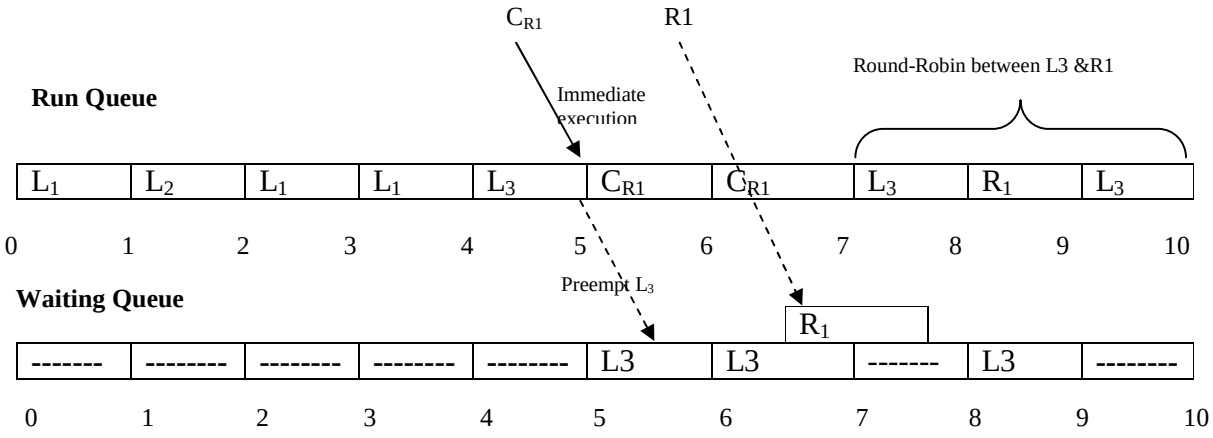


Figure 3. Proposed Scheduling Mechanism for CASE A

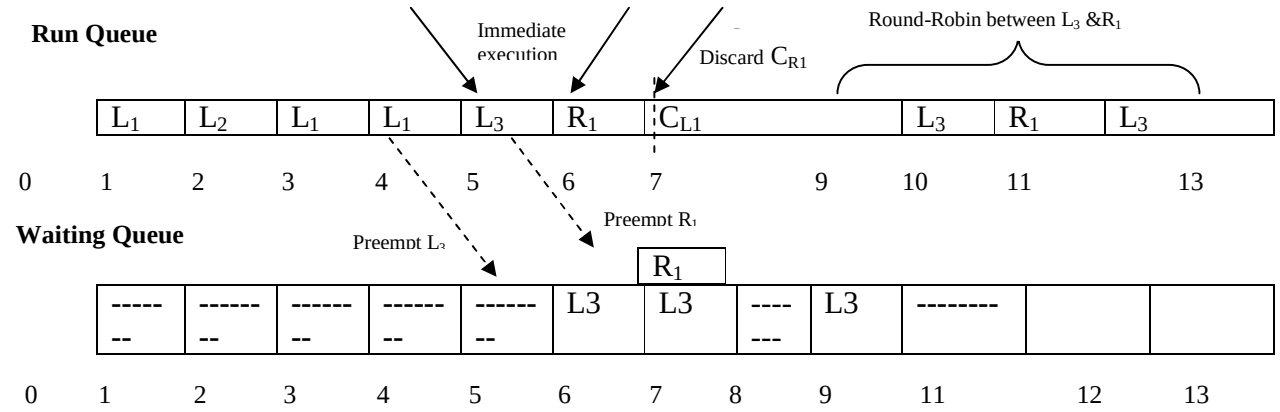


Figure 4. Proposed Scheduling Mechanism for CASE B & CASE C

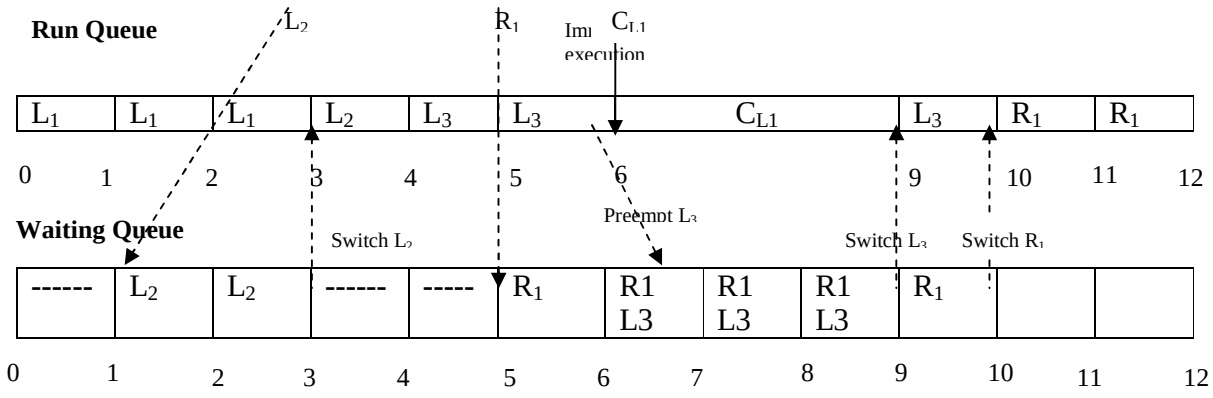


Figure 5. Proposed Scheduling mechanism for CASE D

Live Migration-based Resource Managers for Virtualized Environments:

A Survey

Omar Abdul-Rahman
Graduate School of Information Science & Technology
Hokkaido University
Sapporo, Japan
omar@ist.hokudai.ac.jp

Masaharu Munetomo and Kiyoshi Akama
Information Initiative Center
Hokkaido University
Sapporo, Japan
munetomo@iic.hokudai.ac.jp, akama@iic.hokudai.ac.jp

Abstract— **Virtualization is a technology originally developed for mainframe computing. However, recent developments in the virtualization makes it a key technology to address the problems of modern distributed infrastructure like cloud platforms. Perhaps, one of the most important mechanisms provided by virtualization is the ability to migrate running applications without affecting the end user in a seamless manner. So, virtual machine migration is a promising approach to realize the objectives of efficient, adaptive and dynamic resource manager for virtualized environments. In this the paper, we present the state of art migration based resource managers for virtualized environments, compare and discuss different types of the underlying management algorithms from algorithmic issues standpoint.**

Keywords- virtualization; live migration; management algorithms; consolidation; orchestration

I. INTRODUCTION

Virtualization has attracted considerable interest in recent years, particularly from the datacenters, could platforms and cluster computing communities. By defining an intermediate layer that decouples the operating systems (that is in direct control of the hardware) and the applications, virtualization can address the problems of modern distributed infrastructures like load balancing, high availability, rapid infrastructure deployment and application isolation. Perhaps, the biggest advantage of employing virtualization is the ability to flexibly control resource allocations. The dynamic resource allocation requirements of workload can be satisfied by altering the capacity of a virtual machine at runtime. While, the ability to power-on/off, archive, migrate containers and their workloads enhance the capability of the resource manager to work around resource bottlenecks and faults. [1]

Despite virtualization capabilities, a still challenging question is *how an intelligent infrastructure should optimally maps workload and resource requests onto available virtualized resources utilizing theses capabilities?* It is possible to identify four different trends to realize dynamic resource management systems in virtualized environments. A large part of the literature is based on *request distribution policies* [2][3][4][5][6]. In this trend, a controller adopts a policy that dynamically adjusts requests distribution to share resource among the running applications. By using *virtual machine slicing* [7][8], a controller manages resources by dynamically change virtual machines allocations (or fractions of usage).Then, we have resource management using *virtual machine replication/instantiation* technique [9]. Replication/instantiation entails the creation of a local virtual machine's replica (or instantiate a new virtual machine) in the target physical server. The load would be shared between the two instances, diminishing the stress on the local physical server. Finally, we have resource management by *virtual machine migration*.

The successive developments in the field of virtualization technology greatly reduced downtime overhead associated with migration. For example, support for migrating groups of processes across OSs was presented in [10], but applications had to be suspended and it did not address the problem of maintaining open network connections. In [11] Virtualization support for commodity operating systems led towards techniques for virtual machine migration over long time spans, suitable for WAN migration [12]. More recently, the most two popular modern virtualization technologies products from VMware [13] and Xen [14] have realized the notion of *live* or *seamless* migration of VMs that involve extremely short downtimes ranging from tens of milliseconds to a second. Thus, virtual machine migration is emerged as a promising technique to be utilized by resource management algorithms to rapidly resolve resource allocation problems in the virtualized environments. However, up to date this approach has received a little attention. Thus, we limit the scope of this survey to the dynamic resource manager based on the live migration technique. The remaining part of this paper is organized as the follows. A general live migration based resource manager system is presented in Section II. Different types of underlying management algorithms is presented and discussed in Section III. We conclude the paper and highlight possible directions of future research in Section IV.

II. GENERAL LIVE MIGRATION BASED RESOURCE MANAGER

In this section, architecture is presented that highlights the main phases of processing for a general live migration

 ISBN: 978-1-61208-106-9

based resource manager. There are different levels of virtualization; however, we are care here for operating system level virtualization, which supports the scenario shown in Figure 1; on each physical machine (PM) there is a Virtual Machine Monitor (VMM), also called hypervisor that allows for many virtual machines (VMs) to share the physical resources. A dynamic resource manager imposes a fair resource sharing policy on the competing VMs by producing suitable migration orders, which are implemented by VMM. The widely testified assumption is that each application is represented by a single VM running in a shared hosting model. Exceptions are [15], where each application can be represented by one or more VMs, [16][17] that support the notion of virtual clusters of VMs, while, [18] is designed for multi-tier distributed infrastructure. The resource manger is usually processed in an iterative manner. In each iteration, there are three main phases of processing that can be described as follows:

1. Pre-allocation Phase; the duty of the resource manager here is to collect usage data from the running nodes within a specific measurement interval employing a specific monitoring engine. The details of this engine depend largely on the employed virtualization technology and the required data to be collected. Through these data, the manger can keep a general view about the performance level in the running nodes. It triggers re-allocation if the there is violation(s) for the predefined triggering conditions.
2. Migration Planning phase; this is the most critical part of the processing, since it is the duty of the resource manager to produce a suitable migration plan or orders for a new placement that can eliminate or minimally violate the triggering conditions. The migration plan usually consists of sender PMs, migrated VMs and receiver PMs.
3. Migration execution phase; it is the duty of the VMM to implement the migration plan or orders produce by the resource manager. The specifics of this phase depend on the employed virtualization technology.

The above mentioned architecture is widely used in the literature. It can be described as a *single-tier* architecture, which differs from multi-tier architecture used in [19] or arbitrator architecture used in [20][21].

III. TYPES OF MANAGEMENT ALGORITHMS

In this section, the underlying algorithms that are used to the implement the surveyed live migration based resource manager are discussed. In order to classify the different algorithms properly, *design model* was adopted as classification criteria from algorithmic issues stand point. By design model, we refer to the standard procedure followed by an algorithm to solve a given problem. Taking design model into consideration, it is possible to broadly classify the surveyed algorithms into *ordering algorithms*, *Constraint Programming* (CP) based algorithms and *Genetic Algorithms* (GAs). Moreover, we discuss the differences in *key concepts or techniques* that exist among the same class

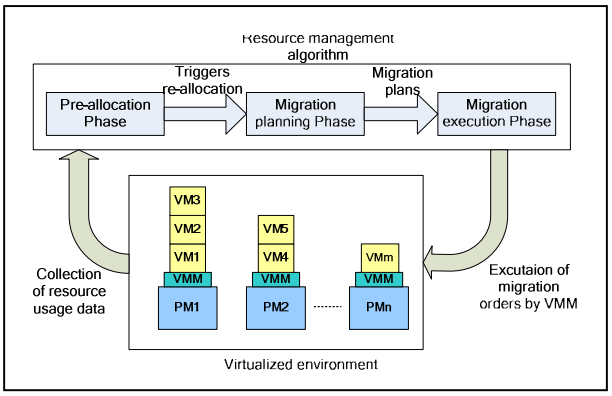


Figure 1. Architecture of a general live migration based resource manager.

of algorithms. These key techniques can be described as an *ad hoc* techniques or modifications that are introduced into a specific algorithm to enhance its performance against the weakness that may exist into the standard procedure to solve a given problem. They give a specific algorithm the unique features and merits against other algorithms. On the other hand, the existed differences among the surveyed algorithms in key techniques do not affect the above mentioned classification into three groups, as long as they follow the same underlying design model. Therefore, the discussion in this section is centered on design model, which give us a brief picture about the processing method, and algorithms features, which give us an idea about the design goals that a specific resource manager is able to satisfy. In addition, experimental results are presented whenever possible. Both experimental results and algorithm features gives an insight about suitability and the ability of a specific resource manager in solving the problems of resource allocation in virtualized environments.

A. *Ordering Algorithms*

VMs resource allocation problem is NP hard problem that is usually formulated as a generalization of knapsack or generalization of its special form of bin packing problem. Therefore, the literature is dominated by ordering algorithms, which is a popular heuristic approach to address such class of problems. The resource manager under such kind of algorithms should resort for a specific ordering model to answer the questions of; From where to migrate? Which VM to migrate? Where to migrate? These questions are the core of the migration plan discussed in Section II. The assignment of a candidate VM to a candidate PM is done only when assignment conditions are met, while, VMM receives migration orders when stopping criteria are met.

The comparison of these algorithms is summarized in Table I, while, these algorithms can be discussed as the follows.

Verma et al., in [22], proposed Dynamic Management Algorithm (DMA) that it is based on Polynomial-Time Approximation Scheme (PTAS) heuristic algorithm. The algorithm operates by maintaining two types of ordering lists, which are migration cost list (VMs on each PM is arranged in non-decreasing order according to their migration cost)

 ISBN: 978-1-61208-106-9

TABLE I. COMPARISION OF ORDERING ALGORITHMS

Resource manager	Design model				
	From where to migrate	*Which VM to migrate*	*where to migrate*	*Assignment conditions*	*Stopping criteria*
DMA [22]	Max-Min thresholds violating PM	Lowest utilization VM (migration cost)	Lowest PM (unused portion of resources)	Resource constraints satisfaction	Repeat migrations until Overload or underutilization elimination
TOPSIS Algorithm [23]	overloaded PM (threshold breaking volume value)	Best VM candidate (migration cost)	Lowest PM (volume)	Resource constraints satisfaction	Repeat migrations until Overload elimination
Andreolini et al., algorithm [24]	Overloaded PMs (CUSUM algorithm)	Highest loaded VM (Sorting algorithm)	lowest loaded PM (Sorting algorithm)	Each receiver must accept one guest VM in a greedy manner	When all guest VMs are assigned to the receivers
MFR algorithm [25]		All VMs are sorted in descending order according to resource demand forecast	Bin packing heuristic is used to map VMs onto the PMs	Resource constraints satisfaction	Construct a new placement that satisfied assignment condition or minimally violated it
pMapper [20] (power manager)	Overloaded PMs (SLA violations)	Lowest VM (application size)	Bin packing heuristic is used to map VMs onto the PMs	Resource constraints satisfaction	Construct a new placement that satisfied SLA requirements
Sandpiper [26]	Overloaded PM (threshold breaking volume value)	Highest VM (VSR)	Lowest PM (volume)	Resource constraints satisfaction	Repeat Until overloading is eliminated by migration or swap
Ruth et al., algorithm [16]		Heuristic is used to determine over and under utilization VMs	Lowest loaded PM in the same domain or other domain	Resource constraints satisfaction in same domain or other domains	Repeat migrations until overloading and underutilization is eliminated
vGreen [19] (MPC balance)	Overloaded PM (threshold breaking nMPC value)	Lowest VM (vMPC)	lower than overloaded PM (nMPC)	Resource constraints satisfaction and system balance	Repeat migrations until MPC violations are eliminated
vGreen [19] (IPC balance)	Overloaded PM (threshold breaking nIPC value)	Lowest VM (vIPC)	A node that has a lower nMPC value than the sender one	Resource constraints satisfaction and system balance	Repeat migrations until MPC violations are eliminated

and residual capacity list (all PMs are arranged in non-decreasing order according to their residual capacity, which reflects the unused portion of resources). One of the features of this management algorithm is the ability to minimize the migration cost. This is achieved by employing a migration cost function that ranks possible new placements. High rankings will be given for those placements as close as from the current one, while, low rankings will be given for far placements or placements that initiates idle PMs. Another feature is the ability to minimize power consumption by detecting underutilization in the managed system. This achieved by using Max-Min thresholds selection model. When the resource usage of a running PM violates a minimum predefined threshold value, the algorithm tries to pack the running VMs as close as possible thus trying to minimize the number of running PMs.

Comparison experiments were held between the proposed DMA and an exact or optimal DMA. The optimal DMA outperforms the exact one only in the in the optimality of solutions, i.e., mapping VMs onto fewer PMs. However, the proposed DMA has a better time performance and minimum migration cost. In addition, the proposed DMA has the ability to adapt the running resources toward workload variability, while, optimal DMA lacks this adaptability merit. The main limitation of this algorithm is the CPU-memory resource model. Under this model it is not possible to judge the performance of the resource manager under network intensive applications.

Tarighi el al in [23] formulate resource allocation problem as a multi-criteria decision problem and employed fuzzy Decision Making Model (FDM) based on TOPSIS techniques. Fuzzy TOPSIS is a technique for ordering preference by similarity to an ideal solution. Adopting this technique gives the authors the flexibility to include beside the usual deterministic information another kind of information in the form of linguistic and fuzzy parameters. For example, like Sandpiper in [26] the authors used *volume* or ordering function to detect the degree of overloading in the running PMs. However, unlike the volume in Sandpiper, which usually processes deterministic data of resource usage (CPU, memory and network), volume here is able to process for example Quality of Service (QoS) as a linguistic parameter and temperature of the PM as a fuzzy parameter. By including the temperature, the manager becomes able to forecast failure for a specific node and takes fault tolerance migration actions. This an advantage compared with other resource managers. Another point of comparison with Sandpiper [26] is VMs ordering functions. Both of them using the same kind of function, however, the function here is included further data of memory footprint size to arrange VMs. That adds flexibility to the performance of the resource

 ISBN: 978-1-61208-106-9

manager by giving priority to specified VMs during migration.

Experimental investigation validates the performance of the system under different critical scenarios. However, the results showed that method suffers from the problem of time-consuming calculations.

Andreolini et al., in [24], proposed management algorithms for cloud computing context. One of interesting part of this algorithm is that of using a selective CUSUM algorithm to detect the critical nodes instead than the widely used threshold violation selection model. An important feature of this algorithm is the ability to capturing significant changes of running PMs load. Thus, limiting the number of sender nodes and avoid needless migrations generated by false alarms due to many instantaneous spikes, non-stationary effects, and unpredictable and rapidly changing load. Another part is using a load trend-based sorting algorithm to select the candidate VMs for migrations and the receiver nodes. An important feature of this algorithm is the ability to clearly classify the trend of the running VMs load to increasing, decreasing, oscillating or stabilizing, Thus, contributes to limiting number of migrated VMs to only few critical ones. Finally, the algorithm distribute the migrated VMs by assigning each receiver only one VM. Thus, we can avoid load imbalance shifting, which is undesirable effect that generates frequent fluctuations negatively impact system performance and stability. The authors confirmed the performance of their system by experiments on traces coming from cloud platforms, however, no experimental details are provided in this paper.

Bobroff et al., in [25], proposed a dynamic server migration and consolidation algorithm introduced as Management-Forecast-Reallocation or MRF algorithm. The interesting part of this algorithm is maintaining a gain formula that it is used to combine online resource measurements with resource demand forecast. This gain formula is the core of VMs ordering list. Using of forecasting technique gives MFR the ability to adapt the available resource to the workload variability in a proactive manner, thus providing probabilistic SLA guarantees. In addition, it can minimize the number of running PMs at the time of decreasing workload behavior thus minimize power consumption. However, the downside of forecasting technique is the increased sensitivity of MFR performance toward remapping interval. For longer remapping intervals, the performance of MFR is degraded. This is showed by the experimental results. Another feature of this algorithm is the ability to deal with the critical scenario of high workload utilizations. When it is impossible to totally eliminate capacity violations, it generates a placement that minimizes violations as much as possible.

The experimental results were based on workload traces across a variety of operating systems, applications, and industries. The results were proved that MFR outperforms static allocation in term of reducing the physical resource consumption for a specific SLA violations rate by 50% and reducing SLA violations at a fixed capacity by 20%. On the other side, the main limitations is in the resource model (CPU only), thus we could not get the full picture of the resource manager performance under memory and network intensive applications.

Sandpiper is a resource manager proposed in [26]. An interesting feature of Sandpiper is that of using a score function or ***volume*** to measure the degree of overloading in the running PMs. This function is designed to capture overloading along three dimensions of CPU, memory and network. In addition, another score function or Volume Size Ratio (VSR) is designed to order VMs in a descending order according to their memory size, thus minimize the migration overhead. Moreover, Sandpiper, like [25], is designed to deal with critical scenario of high workload utilization but using a different approach. Sandpiper first tries to mitigate a hotspot by migration. If failed, it tries to swap a high VSR VM in the overloaded node with one or more of low VSR VMs in the destination node. The experimental results showed that migration overhead is less than that of swapping overhead; however, swapping increases the chances of mitigating hotspots in clusters with high average utilization. Another feature is the ability of Sandpiper to address system stability by avoiding needless, wasteful and thrashing migrations. Sandpiper avoids needless migrations generated by false alarms by triggering migrations only if thresholds or SLAs are exceeded for sustained time. In case of increasing number of hotspots, Sandpiper either implements a partial solution or gives up entirely wasteful migrations. However, monitoring techniques is the most interesting part of the paper. Here, the authors proposed black box and gray box monitoring techniques. In the gray box technique, it is possible for Sandpiper to relay on some OS level statistics beside external usage to infer SLA violations. However, in the black box technique, Sandpiper depends only on the external usage to infer the SLA violations. This ability to use some OS level statistics gives gray box based Sandpiper an edge performance over black box one. Experimental results showed that gray box based Sandpiper behaves proactively. So, it produces fewer swaps, resolve situations faster and balance system more quickly compared with black box Sandpiper. By comparing with static allocation, Sandpiper eliminates all hotspots, while static allocation failed. In addition, Sandpiper reduces the number of intervals experiencing sustained overload by 61%. the experiments showed that the system overhead has insignificant CPU and I/O requirements and has a negligible impact on performance, while, the system can scaled up to 500 VMs with computational complexity of less than 5 seconds. For very large data centers with thousands of VMs, authors proposed that computation could be split up across multiple nodes, or the center's servers can be broken up into pools, each controlled independently by its own control plane. On the other hand, results showed that the quality of Sandpiper degrades for long measurements interval.

Ruth et al., in [16], presented a resource manager for a system called Violin, which composed of virtual network of VMs. An important feature of this manager is the ability to identify and eliminate underutilization by employing max-min threshold violations detecting heuristic. Thus, reduce power consumption. However, the novel part of this paper is in the employing two different virtualization techniques of

 ISBN: 978-1-61208-106-9

VM slicing and VM migration to resolve resource allocation problem. By VM slicing, the manager first performs fine-grained control over per-host memory and CPU allocation utilizing memory ballooning and CPU scheduling techniques provided by VMware and Xen technologies. If failed to resolve resource allocation problem, it adopts the VM migration option. This hybrid approach is an efficient method for limiting the number of migrations, thus minimize migration cost.

The experimental study reported a small migration overhead and concentrated mainly on validating the performance of the system under different critical scenarios. However, the main limitation is in the consideration of only two dimensions of CPU and memory as a sources model.

Dhiman et al., in [19], presented vGreen, a multi-tiered software system, for energy efficient computing in virtualized environments. The innovative part of this system is in the using of novel hierarchal parameters processed in a novel multi-tier architecture. The authors developed the idea of the hierarchal parameters from experiments on benchmarks from SPEC-CPU 2000 suite, namely *mcf* and *perl*. Experiential results showed that co-scheduling of VMs with heterogeneous characteristics on the same PM is beneficial from both energy efficiency and performance point of view. In order to capture these characteristics, authors developed two kinds of parameters which are memory accesses per cycle (MPC) and instructions per cycle (IPC). These parameters are maintained in two *hierarchal* levels of node (nMPC, nIPC) and VM (vMPC,vIPC). On the other hand, the multi-tier architecture is shown in Fig. 2. In contrast to the single tier architecture that it is shown in Fig. 1, where the migration planning phase is implemented in a single-tier or step, the migration phase in multi-tier architecture is implemented in four sequential steps. Both of the above mentioned architecture and parameters gives vGreen the merit of addressing performance and power balancing requirements more accurately compared with other systems presented in this survey. Therefore, when the manager implements MPC balance tier (Table I), this results in a better overall performance and energy efficiency across the cluster. While balancing of IPC metric values in the IPC balance tier (Table I) results in better balance of power consumption across the PMs. The performance goals are further checked in the Uitl balance tier which eliminates overcommitted nodes from the cluster. Finally, VM consolidation tier contributes in the resource adaptation ability of the manager by eliminating underutilization from the cluster. By comparing vGreen with VM scheduler that mimics the Eucalyptus, which is a state of art strategy for cloud context, and under heterogamous workload conditions, the experimental results showed that vGreen outperforms along author`s developed metrics of energy savings, which captures energy consumption reduction, Weighted Speedup, which captures migration overhead and Reduction in Power Imbalance, which captures the power consumption variance within the machines of the cluster. An overall performance and system level energy savings by 20% and 15% improvement were achieved. However, under homogenous workload condition both performed the same. On other hand,

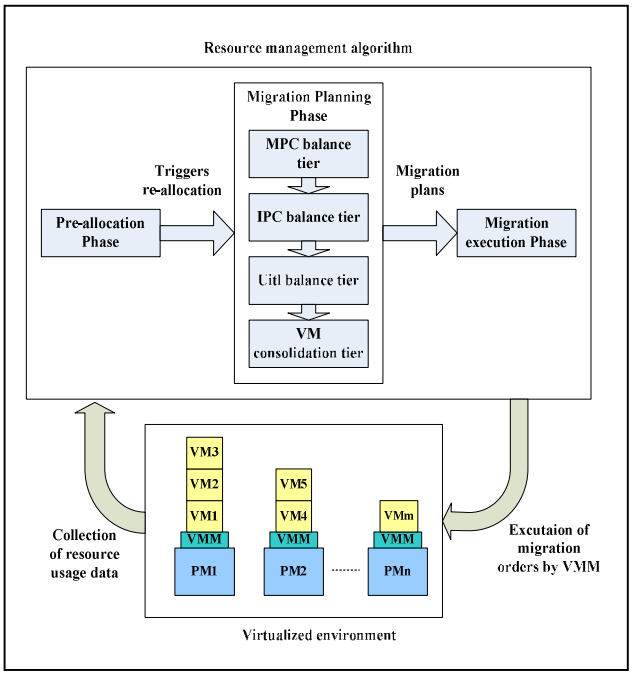


Figure 2. Multi-tier architecture of vGreen system.

the limitation of this manager is in the two dimensions CPU-memory resource model.

In [20], Verma et al. propose the power aware application placement framework or pMapper as the resource manager. The novelty in this resource manager is the architecture which consists of arbitrator that issues migration orders based on the information communicated by performance, power and migration managers. This novel architecture gives pMapper the ability to serve many of management objectives at the same time. The system operation can be described as follows. The performance manager continuously checks the performance level in the running nodes against the performance targets specified by SLA. The Power manager utilized an experimental developed CPU based power model in the generation of power-minimizing new placements (as shown in Table I). While, the migration manager utilized an experimental developed migration cost model that quantify migration cost from the decrease in throughput because of live migration and estimate the revenue loss because of the decreased performance (as given by SLA). Finally, the arbitrator constructs a new placement by picking up the optimal migrations that trade-offs power-migration targets and achieve SLA goals.

In [21] the authors modified their pMapper. The core of their modifications is proposing a new power models that related power consumption to the CPU, memory footprint and caches usage characteristics of the application. The essence of their new proposal is to sort all applications in ascending order by their memory usage. Then, classify them to three categories. Category 1 or small applications can be packed together respecting memory and CPU limits. Thus, avoid performance degrade. Category 2 or large applications can be packed only based on the CPU limit. Thus, achieve

 ISBN: 978-1-61208-106-9

maximum power savings. Finally, category 3 or medium applications can be packed either as category 1 or category 2 applications.

The experiments were performed to compare the new CPU-Cache based pMapper with the old CPU based pMapper. Both algorithms succeed in reducing the number of the running nodes from 11 to 4. However, CPU-Cache based pMapper outperforms in term of performance overhead. Then CPU-Cache based pMapper is compared with cache-oblivious strategies. For CPU-Cache based pMapper only Category 3 applications have performance impact, while for cache-oblivious strategies more than half of the applications have performance impact. However, this CPU-memory resource model is still limited since it ignores the effect of network dimension of the resources.

B. Constraint Programming

The main idea of the constraint programming based resource manger is to formulate the VM resource allocation problem as a constraint stratification problem then applies a constraint solver to solve the optimization problem. The ability of this solver to find the global optimum is the main motivation to adopt this approach. In the literature, we have identified two papers. The comparison is presented in Table II, while, the discussion can be presented as follows.

Entropy resource manager [27] utilizes Choco constraint solver to achieve the objectives of minimizing the number of the running nodes and minimizing migration cost. The operation of the algorithm can be described as follows. Entropy iteratively checks optimality constrain, i.e., the current placement uses minimum number of the running nodes. If Entropy at VM packing problem (VMPP) phase success in constructing a new optimal placement (uses fewer running nodes), it will activate the re-allocation. In addition, Entropy employs a novel experimental developed migration cost model that relates memory and CPU usage with migration context. High parallelism migration steps reduce the cost, while sequential and infeasible migration steps increases cost. Using of constraint programming techniques facilitate the task of capturing such context. However, considering only viable processing nodes (a running node can support only a single active VM, while other co-allocated VMs should be inactive) and CPU-Memory resources model are the limitations of Entropy model.

In order to speed up the computation process that can be expensive, authors used many optimization techniques. Some of the techniques used in the VMPP phase are used to detect and exclude partially constructed solutions as soon as possible if they violate the optimality and viability constraints. Others used to reduce search space, by limiting the search for the promising region near from the optimal value by imposing upper and lower bounds. In addition, the authors devise a metric of the number of active VMs divided by the number of nodes to calculate the lower bound, while the upper bound is identified by using First Fit Decreasing (FFD) heuristic. Finally, authors devised equivalence classes metric (VMs memory size to the CPU states), which is exploited to reduce the size of search tree. Moreover,

TABLE II. COMPARISION OF CONSTRAINT PROGRAMMING ALGORITHMS

Resource manager	Design model	Constraint Solver phases of processing	
Entropy [27]	Initialization	Optimality and viability constraints violations.	
	Processing	VM packing problem	
		VM replacement problem	
	Stopping criteria	Can be aborted at any time	
Van et al., algorithms [15]	Initialization	GDM receives inputs from LDM and monitoring probe	
	Processing	GDM	VM Provisioning
			VM Packing and replacement
	Stopping criteria	Can be aborted at any time	

Equivalence classes are also exploited as an optimization technique in VM replacement problem phase with more strict constraints.

Scalability experimental results showed that the system complexity is directly related to the characteristics of running VMs and the underlying PMS. More computation time is required for configurations sets that have many VMs memory requirements and many CPU states. Moreover, the scalability of the system is showed to be related to number of VMs per node. Higher the ratio, longer the time required to compute a solution. On the other hand, when compared with First Fit Decreasing heuristic (FFD), Entropy outperforms in term of minimizing the number of unsatisfied VMs and producing reconfiguration plan with better cost. In addition, Entropy outperforms static allocation by 50% and FFD by 25% in term of minimizing the number of running nodes over a collection of NASGrid benchmarks.

Van et al., in [15], proposed an architecture and management algorithms for cloud computing contexts. The management algorithms are based on Entropy resource manager [27]. However, the main advantage over Entropy resource manager is in the novel architecture that separate the management decisions among a Local Decision Module (LDM) associated with each application and a Global decision Module (GDM). The ability of the resource manager here to combine and automated application provisioning problem with resource adaptation problem is the outcome of this architecture. Another advantage is in the including of operations costs in the migration planning model. However, like Entropy the resource model is limited to the CPU-memory dimensions. The architecture and the algorithms are validated through simulation experiments. The attempt is still in the early development phase.

C. Genetic Algorithms

GAs are metaheurstics that are inspired by evolutionary biology. It starts the evolution process from a population of initial solutions and changes them very fast by applying the usual selection and recombination operators. The rate of change reduces gradually when we reach the optimal solution. This incremental behavior, besides the simplicity of implementation and the ability of parallelization makes GAs a promising approach for VMs resource allocation

 ISBN: 978-1-61208-106-9

problem. In the literature, we have identified two papers. Table III presents and compare the design model for these algorithms, which can be discussed as follows.

Campegiani in [18] used GA to find the optimal allocation of virtual machines in a multi-tier distributed environment. The innovative part of this work is the formal model that addresses beside the usual quantitative resources (CPU, memory and network), qualitative resource like physical position awareness. In addition, this model allows for multiple-SLA representation for each VM. This scheme gives the proposed GA the ability to capture multi-tier distributed infrastructure, by a signing a profit for specific SLA. Maximize the profit through evolution is translated into maximization of physical resources efficiency, while accommodating for transient workload surges. However, the challenge for GA comes from the infeasible solutions that can be appeared as a byproduct of the evolution process. The author addressed this challenge by devised penalty function that differentiates feasible solutions, while penalize unfeasible ones. These infeasible solutions are fixed using a repair operator. Simulation Experiments on arbitrary dataset were held to validate the algorithm which is still in the early development phase.

Nakada et al., in [17], implemented a prototype Virtual Machine packing optimization mechanism on Grivon, which is a virtual cluster management system for Hardware as a Service (HaaS) cloud system type. The most interesting part of this algorithm is the penalty function with the objective of packing VMs tightly onto the PMs to minimize the number of the running nodes, while attempting to minimize migration cost and respecting SLA performance levels at same time. Experiments were held to validate the performance of the system. According to the authors, experimental results showed that the GA based approach is flexible and fast enough for VM packing problems and represent a promising approach.

TABLE III. CPMPARISIOON OF GENETIC ALGORITHMS

The design model of GA	the resource manager	
	Campegiani [18]	Nakada et al [17]
Chromosome representation	Binary representation	Integer sequence
Initial population	Applying First fit, next fit and best fit heuristic on the input configuration.	Repeatedly applying mutation on the input configuration.
Selection operator	tournament selection scheme	regular normalized weighted roulette method
Crossover operator	Uniform crossover	One point cross over
Mutation operator	Fix rate mutation operator	Mutations will cause real change in the individual by reordering two randomly chosen numbers.
Type of fitness function	Penalty function	Penalty function
Replacement scheme	Inject new individual, remove lowest fitness one.	
Special operators	Repair operator	

IV. CONCLUSION AND FUTURE RESEARCH

Virtualization is a re-emerged technology that offers powerful resource management mechanisms that can cope with the challenges of modern distributed infrastructures like datacenters and cloud platforms. Live migration based resource management systems is a promising approach for efficiently manage resources and rapidly eliminate hostpots in the virtualized environments. In this paper, we surveyed state of art resource managers describing them using general resource manager architecture and presenting different types of the resource management algorithms which classified to ordering, constrain programming and genetic algorithms. We compared and discussed these algorithms from the design model and key concepts and techniques standpoints.

It is possible to highlight the below mentioned fully unexploited recent trends; we believe that they will attract a greater attention in the future directions of research by developing more formal models, trying alternative approaches, devising metrics or performing feasibility experiments.

- Qualitative resources are an important recent trend, which gives a resource manager the merit to handle qualitative requirements of resource allocation. For example, physical position awareness, which is described as a qualitative resource, is an important requirement for multi-site virtual clusters. In the survey, we have identified only one paper [18], which is still in the early development phase, which addresses such theme.
- Developing a resource manager that has the ability to combine on-the-fly (or dynamic) application provisioning with dynamic application consolidation at the same time is an important recent trend especially for cloud contexts. We have identified in the survey only one paper [15], which still in the early development stage, that address such theme.
- Developing a hybrid resources manager that uses more than one virtualization techniques (like VM migration and VM slicing) is an important trend that can combine the benefit of both techniques. VM slicing can contributes into migration cost limitation or minimization. In addition, it is useful to capture the structure of modern multi-domain or multi-tier infrastructures. On the other hand, VM migration, that finds global solutions for resource allocation problems, can enhance the performance of VM slicing technique, which is useful in finding local solutions for resource allocation problem. We have identified in the survey only one paper [16] that apply this method for a grid computing platform.
- According to the survey, ordering and CP approaches are suffers from time-consuming calculations. It requires innovative techniques and complex architectures to overcome this difficulty. We believe that the future research developments will turn toward more robust and faster metaheuristic algorithms. Multi-objective GA is a possible

 ISBN: 978-1-61208-106-9

promising candidate algorithm for future cloud systems.

- Hybriding CP and GA approaches can be seen as a possible direction of future research. The combined approach can benefit from their different aspects. From one side, constraint programming is a good approach to model complex constraints. On the other side, GA can greatly fasten CP expensive processing.
- Dataset is important for algorithm development. It is usually used to test algorithms against it. It have been noted from the surveyed papers that many researchers resort to the randomized configurations for developing their algorithms. This field of research lacks suitable dataset that captures virtualized infrastructure. Therefore, future development in this field will be largely affected by the development of suitable dataset by specialized community like operational research or integer programming.
- Finally, future development in this field is largely related to simulation and experimentation procedures. It has been noted from the surveyed papers, that there is a need to define standard benchmarks applications, standard metrics to measure the goodness of the resource managers and to perform experimental tests under large configurations of VMs and PMs and comparing among different states of art resource managers.

ACKNOWLEDGMENT

First author is supported by the Japanese Government (Monbukagakusho) Scholarship program to complete his PHD study. In addition, I would like to thank the anonymous for their efforts, which contributes much to the development of this paper.

REFERENCES

[1] C. Clark, K. Fraser, A. Hand, J. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, "Live Migration of Virtual Machines," In Proc. of the 2nd conference Symposium on Networked Systems Design and Implementation, vol. 2, pp. 273 – 286, 2005.

[2] K. Appleby, S. Fakhouri, L. Fong, M. Goldszmidt, S. Krishnakumar, D. Pazel, J. Pershing, and B. Rochwerger, "Oceano - SLA-Based Management of a Computing Utility," In Proc. of IFIP/IEEE Symposium on Integrated network management, Seattle, WA , USA, pp. 855 – 868, May 2001.

[3] J. Chase, D. Anderson, P. Thakar, A. Vahdat, and R. Doyle, "Managing Energy and Server Resources in Hosting Centers," In ACM SIGOPS Operating Systems Review, vol. 35, issue 5, pp. 103 – 116, December 2001.

[4] B. Urgaonkar, P. Shenoy, A. Chandra, and P. Goyal, "Dynamic Provisioning for Multi-Tier Internet Applications," In Proc. of the 2nd IEEE International Conference on Autonomic Computing, Seattle, WA , June 2005.

[5] K. Rajamani and C. Lefurgy, "On Evaluating Request-Distribution Schemes for Saving Energy in Server Clusters," Proc. IEEE International Symp. Performance Analysis of Systems and Software, pp. 111 – 122, 2003.

[6] A. Karve, T. Kimbrel, G. Pacifici, M. Spreitzer, M. Steinder, M. Sviridenko, and A. Tantawi, "Dynamic Placement for Clustered Web Applications," In Proc. of the 15th international conference onWorld WideWeb, Edinburgh, Scotland, pp. 595 - 604, 2006.

[7] J. Almeida, V. Almeida, D. Ardagna, C. Francalanci, and M. Trubian, "Resource Management in the Autonomic Service-Oriented Architecture," In Proc. of International Conference on Autonomic Computing,pp. 84-92, 2006.

[8] P. Padala, K. G. Shin, X. Zhu, M. Uysal, Z. Wang, S. Singhal, A. Merchant, and K. Salem, "Adaptive Control of Virtualized Resources in Utility Computing Environments," In EuroSys '07: Proceedings of the 2nd ACM SIGOPS/EuroSys European Conference on Computer Systems, Lisbon, Portugal, pp. 289 – 302, 2007.

[9] X. Wang, D. Lan, G. Wang, X. Fang, M. Ye, Y. Chen and Q. Wang, "Appliance-based Autonomic Provisioning Framework for Virtualized Outsourcing Data Center. Autonomic Computing," Fourth International Conference on Autonomic Computing, Jacksonville, Florida, USA, pp. 29 – 29, 2007.

[10] S. Jones, A. Arpaci-Dusseau, and R. Arpaci-Dusseau, "Geiger: Monitoring the Buffer Cache in a Virtual Machine Environment," In Proc. ASPLOS' 06, pp. 13-23, October 2006.

[11] K. Govil, D. Teodosiu, Y. Huang, and M. Rosenblum, "Cellular Disco: Resource Management using Virtual Clusters on Shared Memory Multiprocessors," In Proc. SOSP'99, Dec.1999. pp. 154-169.

[12] C. Sapuntzakis, R. Chandra, B. Pfaff, J. Chow, M. Lam, and M. Rosenblum, "Optimizing the Migration of Virtual Computers," In Proc. of the 5th symposium on Operating systems design and implementation, Vol. 36 , Issue SI, pp. 377 – 390, 2002.

[13] VMware, http://www.vmware.com. Last access on 2010.08.22.

[14] Xen, http://www.xen.org, Last access on 2010.08.22.

[15] H. Van and J. Menaud, "Autonomic Virtual Resource Management for Service Hosting Platforms," In Proc. of the 2009 ICSE Workshop on Software Engineering Challenges of Cloud Computing, pp. 1-8, 2009.

[16] P. Ruth, J. Rhee, D. Xu, R. Kennell, and S. Goasguen, "Autonomic Live Adaptation of Virtual Computational Environments in a Multi-Domain Infrastructure," In Proc. IEEE ICAC '06, pp. 5 – 14, June, 2006.

[17] H. Nakada, T. Hirofuchi, H. Ogawa and S. Itoh, "Toward Virtual Machine Packing Optimization Based on Genetic Algorithm", Proceedings of the 10th International Work-Conference on Artificial Neural Networks: Part II: Distributed Computing, Artificial Intelligence, Bioinformatics, Soft Computing, and Ambient Assisted Living, Salamanca, Spain, pp. 651 – 654, 2009.

[18] P. Campegiani, "A Genetic Algorithm to Solve the Virtual Machines Resources Allocation Problem in Multi-tier Distributed Systems," Second International Workshop on Virtualization Performance: Analysis, Characterization, and Tools (VPACT'09), Boston, Massachusett, April, 2009.

[19] G. Dhiman, G. Marchetti, and T. Rosing, "vGreen: a System for Energy Efficient Computing in Virtualized Environments," In Proceedings of the 2009 International Symposium on Low-Power Electronics and Design (ISLPED'09), pages 243-248, New York, NY, USA, 2009.

[20] A. Verma, P. Ahuja, and A. Neogi. "pMapper: Power And Migration Cost Aware Application Placement in Virtualized Systems," In Proc. of 9th ACM/IFIP/USENIX International Conference on Middleware, Leuven, Belgium, pp. 243-264, 2008.

[21] A. Verma, P. Ahuja and A. Neogi, "Power-Aware Dynamic Placement Of HPC Applications," In Proc. of the f the 22nd annual international conference on Supercomputing, Island of Kos, Greece, pp. 175-184, 2008.

[22] G Khanna, K Beaty, G Kar, A Kochut, "Application Performance Management in Virtualized Server Environments," Network Operations and Management Symposium 2006 NOMS 2006 10th IEEEIFIP, pp. 373 – 381, April, 2006.

[23] M.Tarighi, S.A.Motamedi and S.Sharifian, "A New Model for Virtual Machine Migration in Virtualized Cluster Server Based on Fuzzy

 ISBN: 978-1-61208-106-9

Decision Making," Journal of Telecommunications, vol. 1, issue 1, pp. 40-51, Feb. 2010.

[24] M. Andreolini, S. Casolari, M. Colajanni and M. Messori, "Dynamic Load Management of Virtual Machines in a Cloud Architectures," First International Conference on Cloud Computing (ICST CLOUDCOMP2009), Munich, Germany, October 19-21, 2009.

[25] N. Bobroff, A. Kochut and K. Beaty, "Dynamic Placement of Virtual Machines for Managing SLA Violations," IEEE/IFIP International Symposium on Integrated Network Management (IM), Munich, Germany, pp. 119 – 128,May 21-25, 2007.

[26] T. Wood, P. Shenoy, A. Venkataramani, M. Yousif, "Black-Box and Gray-Box Strategies for Virtual Machine Migration," 4th USENIX Symposium on Networked Systems Design USENIX Association & Implementation, pp. 229–242, 2007.

[27] F. Hermenier, X. Lorca, J. Menaud, G. Muller and L. Lawall, "Entropy: a Consolidation Manager for Clusters," In proc. of the 2009 International Conference on Virtual Execution Environments (VEE'09), Washington, DC, USA, pp. 41-50, Mar. 2009.

 ISBN: 978-1-61208-106-9

CrossBit: A Multi-Sources and Multi-Targets DBT

Yindong Yang, Haibing Guan, Erzhou Zhu, Hongbo Yang, Bo Liu
School of Electronic, Information and Electrical Engineering
Shanghai Jiao Tong University
Shanghai, China
{*yasaka, hbguan, ezzhu, yanghongbo819, boliu*}*@sjtu.edu.cn*

***Abstract*—Dynamic binary translator (DBT) is typically used for software migration or binary code optimization. In this paper, we describe the design and implementation of a multi-sources and multi-targets DBT–CrossBit, which aims at fast migrating existing executable source code from one platform to another alien target platform with lower cost. In order to support code translation among multi-sources and multi-targets better, a new intermediate instruction set–VInst, which is independent of any specific machine instructions, has been introduced. Unlike many other existing DBTs which directly translate the binary code of one instruction set architecture (ISA) to another ISA, CrossBit first converts source binary code to VInst specifications, and then transforms them into target platform code, using a granularity of a basic block (BB) as the unit of translation. Additionally, to address the performance issue, we adopt several generic optimization methods to optimize the translated code. Finally, our experimental result indicates that, for the SPECint2000 benchmarks, CrossBit's performance is pleasant and can meet the design requirement.**

Keywords-DBT; intermediate instruction; CrossBit; basic block;

I. INTRODUCTION

Cloud computing, a relatively romantic term, builds on decades of research in virtualization, distributed computing, grid computing, utility computing, and so on. For cloud computing becomes wildly popular and has the potential to transform a large part of the IT industry, developers with innovative ideas inevitably take it into account. As a new evolution of on-demand information technology services and products, cloud computing has become popular until IBM and Google announced a collaboration in this domain in October 2007 [19]. Thus, there are still lots of obstacles to the growth of cloud computing. One of them is performance unpredictability, which caused by the use of virtual machines (VM). Cloud computing moves data and computing away from desktop and portable PCs into large data centers. It involves applications based on different instruction set architectures (ISA), as well as different hardware platforms. VMs have proved to be ideally suitable for the needs of the heterogeneous computing. Not surprisingly, VMs are likely to be common at multiple levels of the data center or server farm.

Virtualization is a core technology for enabling cloud resource sharing. To a large extent, cloud computing will be based on virtualized resources. In the recent 30 years, the computing machinery technology, both hardware and software, has been developing at a tremendous pace. On one hand, the processing speed of processor gets faster, and accordingly there needs less time to perform each certain unit task. On the other hand, the framework of the processor (e.g., *x*86, MIPS, PowerPC) also becomes multiplex in order to satisfy various requirements and be applicable in different scenarios. Different processors usually base on different ISAs. This leads a problem, one kind of processor can only support one appointed or specific operating system (OS) and applications without virtualization technology. However, due to historical reasons, some of these legacy processor architectures (like *x*86) fail to comply with classical virtualization criteria or hardware support. Though it is quite a challenge to migrate existing OSes or applications running on one platform to another platform, binary translation overcomes the obstacles and successfully improves migrating efficiency.

Binary translation technology proposes a transparent and inexpensive approach to migrate applications or OSes compiled for one processor to another. Binary translation can be implemented in either static or dynamic way, more technical detail can be found in [17]. Binary translation techniques are still in infancy compared with their compiler counterparts, plus many binary translators have been proved that they were handcrafted from scratch. For example, commercial binary translators are always closely bound to the underlying machine, or cannot generate code for more than one source and target machine pair, and hence it is difficult to reuse. To solve this problem, this paper proposes a new DBT–CrossBit, which takes "multi-sources" and "multi-targets" as its design goal. With the help of CrossBit, the applications compiled for a specific processor can run on different processors easily, without bringing too much overhead.

In the academic and commercial fields, many binary translators have achieved some success. In terms of improving the performance of applications, the original HP Dynamo system [2] is a dynamic software optimization system that is capable of transparently improving the performance of a source instruction stream as it executes on the native processor. Microsoft dynamic software optimization system Mojo [3], is capable of handling a wide range of programs, including multi-threaded applications that make use of exception handling. In the aspect of migrating applications, Digital FX!32 provides fast transparent execution of 32-bit *x*86 applications on Alpha systems running Windows NT [1]. IA-32 Execution

 ISBN: 978-1-61208-106-9

layer is designed to support IA-32 applications on Itanium-based systems [16]. The UQBT (University of Queensland Binary Translator) [4], is a reusable, component-based binary-translation framework which lets engineers quickly and inexpensively migrate existing software from one processor to the other. Other translators are designed to save power, like Transmeta's Code Morphing technology which is used for unmodified Intel IA-32 binaries to run on the low-power VLIW Crusoe processor [5]. More discussions on extensive prior work can be found in Altman et al. [6], [7].

Different from binary translators mentioned above, CrossBit doesn't translate source code into specific target code directly. Instead, CrossBit dynamically translates one BB of source code into VInst specifications and then into a BB of target code each time. A BB refers to a block of contiguous instructions, starting at the first instructions of the executable file or the instruction executed immediately after the end of last block, and ending with a branch or jump instruction. In particular, it is to be noted that using intermediate instructions bring many benefits. First, it becomes easier to add other source ends and target ends. Second, the workload and complexity of development can be reduced. For example, to translate n kinds of different source ISAs to n kinds of different target ISAs, theoretically, the complexity can be reduced from $O(n*n)$ to $O(n)$, as depicted in Figure 1. Third, intermediate instruction blocks can offer better opportunities for optimization and thus make optimization easier.

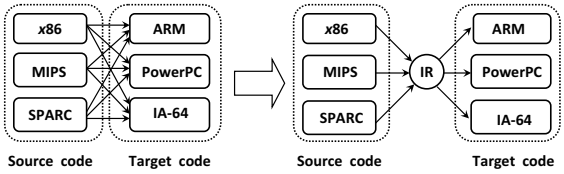


Figure 1. traditional translation model vs. CrossBit's translation model

In this paper, our aim is to to develop a multi-sources and multi-targets DBT with reasonable performance. Specifically, main technical contributions are as follows:

- We design a new DBT–CrossBit, which translates binary code based on different ISAs to other ISAs conveniently;
- We design a new intermediate instruction set–VInst for CrossBit, which involves no specific machine instructions;
- We give a comprehensive experimental evaluation of CrossBit for translating MIPS to *x*86, and MIPS to POWER.

The rest of this paper is organized as follows. Section II describes the framework of CrossBit and how it works in detail. Section III focuses on the design of intermediate instructions. Section IV discusses some optimization tactics adopted by CrossBit. Section V discusses the main issues relevant to our approach to DBT. Section VI presents the preliminary performance of CrossBit, using SPECint2000 as our benchmarks. Section VII gives the summary of this paper and the future research on DBT.

II. The Framework of CrossBit

Dynamic binary translation is an attractive technology for running legacy applications and OSes on the platforms that the software is not originally compiled for. However, the dynamic binary translation technology itself is very complex and difficult to implement. For one thing, developing a complete application-level translator from scratch always takes a lot of manpower and material resources, let alone the development of a system-level translator [18]. Moreover, if one wants to run binary code of one popular platform (like *x*86) on a less popular platform (still in use currently), e.g., SPARC, PowerPC, and MIPS, he has to develop a translator for each of these platforms. In contrast, it is appreciated if there is a translator that can add source ends or target ends easily without repeating the development work. CrossBit is such a translator, and the design goal of CrossBit is retargetable and extensible. We hope that CrossBit will be a promising and reliable basic research platform for studying application-level DBT in the future.

We divide the framework of CrossBit roughly into three parts:

- *Front end*: loads binary executable code into memory and transforms the source binary code into VInst specifications;
- *Intermediate layer*: forms the VInst specifications to blocks and realizes optimization;
- *Back end*: transforms the intermediate blocks to target instructions and executes them immediately.

Figure 2 shows a high-level view of CrossBit. The rectangular boxes related to front end and back end, while the dotted boxes belong to the intermediate layer. We first give a brief overview of all the main components, and then explain how they interact with each other.

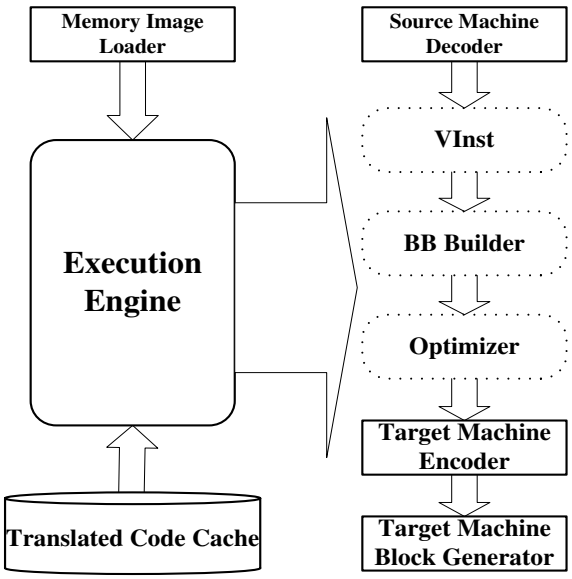


Figure 2. The framework of CrossBit

The names and functions of these components are given as follows:

- *Memory Image Loader*: loads the source binary code into the memory of the target platform, and maps the guest application address space to the host virtual address space;
- *Source Machine Decoder*: analyzes and decodes the source binary code, and converts it to VInst specifications;
- *BB Builder*: constructs VInst specification blocks;
- *Optimizer*: optimizes VInst specification blocks;
- *Target Machine Encoder*: encodes intermediate instruction blocks to target machine instructions;
- *Target Machine Block Generator*: manages SPC (Source Program Counter) and TPC (Target Program Counter) in a hash table and updates their relationship when necessary, so as to speed up the lookup process of target blocks;
- *Translated Code Cache*: caches the translated code so as to save the time of retranslation;
- *Execution Engine*: the commander of CrossBit, in charge of scheduling all components.

The workflow of the entire CrossBit system is as follow:

1) Get the SPC of the first instruction in the next BB, and check whether its corresponding TPC exists in hash table;
2) If the TPC exists, indicating that BB has been already translated and cached, jumps to that TPC and executes that BB directly; Otherwise, turn to 3);
3) *Source Machine Decoder* decodes the following source instructions until the last BB is met, where the decoded instructions will be the input to *BB Builder*;
4) *BB Builder* constructs intermediate instruction blocks;
5) *Optimizer* conducts optimization on the intermediate instruction blocks, e.g., elimination of redundant load instructions, BB linking and so on;
6) *Target Machine Encode* encodes intermediate instructions to target machine instructions which are cached in *Translated Code Cache*, and then *Target Machine Block Generator* updates hash table.

Execute Engine executes 1)∼6) repeatedly until the end of the program. CrossBit takes one BB at a time, keeps doing the job of "translate-execute" cycle, as shown in Figure 3.

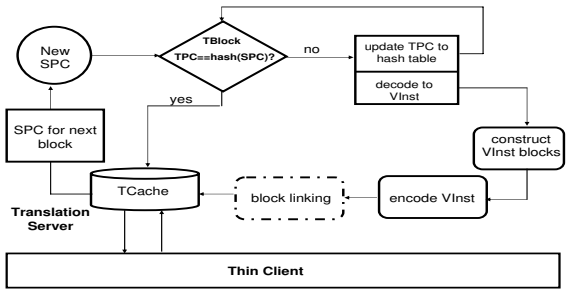


Figure 3. The workflow of CrossBit

III. INTERMEDIATE INSTRUCTION SET

The challenge of designing an intermediate instruction set is how to provide enough high-level run time information about program behavior, as well as be appropriate as an architecture interface for all external applications, libraries, and operating systems. There are many virtual machines using intermediate instructions as a transition layer, such as JVM and Microsoft's Common Language Infrastructure (CLI). However JVM and CLI have some limitations. Both of them use complex, high-level operations with large runtime libraries, and they are tailored for object-oriented languages with a particular inheritance model and their complex runtime systems require significant OS support in practice [15].

An intermediate instruction set can include rich program information for optimization, and can be independent of most implementation–specific design choices, but it is not suitable for a certain hardware implementation. Take LLVA [15] for example, LLVA is a good intermediate instruction set for C++ high level language, but LLVA not suitable for CrossBit to handle low level binary code. Of course, we know that it's impossible to design a universal intermediate instruction set for all conceivable architecture designs. Instead of designing VInst (Virtual Instruction) as the intermediate instruction set that is suitable for DBT to handle low level binary code. VInst must be designed to be regular and simple, for the reason that source instructions and target instructions are all low level machine instructions, and the performance of CrossBit is sensitive to the cost of translation. In a sense, VInst is a kind of low-level ISA. It is similar to RISC ISA and has main characteristics as follows:

- unlimited numbers of 32-bit virtual registers, marked by V_n, wherein the value of V_0 always returns zero;
- "*Load-Store*" style, that is only 'load' or 'store' instruction can access the memory;
- base plus displacement is the only addressing mode;
- instructions of VInst only exist in memory;
- every instruction of VInst is orthogonal, in other words, each instruction can't be replaced by other VInst instructions.

The design of VInst affects the quality of generated target code, and therefor affects the efficiency of CrossBit. When designing VInst, we seek the balance between the performance and the cost. On one hand, VInst is kept as simple as possible so as to reduce the cost of translation. On the other hand, the sematic of VInst should be rich enough to support various characteristics of different ISAs. Thus, by studying the design of some popular ISAs, we have picked up 27 most commonly used instructions to compose VInst. The translation effort is in combing VInst instructions into more complex specific machine instructions.

VInst comprises six kinds of basic instructions which

 ISBN: 978-1-61208-106-9

are compatible with most popular ISAs. They include arithmetic/logical, control transfer, data transfer, memory access, register state mapping and special instructions. Every kind of instructions is shown in Table I.

Table I
THE MOST COMMONLY USED IRS

Type	Instruction Name
register state mapping	GET PUT
memory access	LD ST
data transfer	MOV LI
arithmetic/logic	ADD SUB AND NOT XOR OR MUL MULU DIV DIVU SLL SRL SRA CMP SEXT ZEXT
control transfer	JMP BRANCH
special	HALT SYSCALL CALL

According to the original version of CrossBit we developed, CrossBit translates binary code compiled for PISA (Portable Instruction Set Architecture) into *x*86. That is, CrossBit translates binary PISA code into *x*86 instructions. CrossBit reads the memory at the address indicated by SPC to produce intermediate representation objects, where each intermediate representation object represents one source instruction, not binary representation actually. The translation from a source instruction to VInst takes three steps:

- via GET instruction, mapping source machine registers which the PISA instruction requires to read to virtual registers;
- use one or more VInst instructions to implement the function of each PISA instruction;
- via PUT instruction, mapping the result from virtual registers to source machine registers.

So far, we have used GCC compiler to generate binary executable file. A simple "hello world" C++ source program leads almost 20 thousands of VInst instructions, the details are omitted herein for brevity it's impractical to present a whole example here. A key issue of translation is semantic matching. Instruction swelling is inevitable because the translation policy we adopt here is to be correct first then better. Thus, there needs later optimization to offset the cost.

IV. OPTIMIZATION

Binary translation always serves as an alternative way to execute legacy software. The performance of the translated code should be competitive with the legacy architecture's performance. However, overhead is inevitably lost during the process of translation, because the legacy software has the luxury of being produced using an optimizing compilation. In lack of high-level language code, binary translators cannot perform available optimizations compared to compilers. In this case, optimization is particularly important to dynamic binary translation. One common approach to improving binary translation performance is profile-guided optimization. Profiling is a process for dynamically collecting program information (instruction and data statistics) that is used to guide optimization during the translation process. As a DBT, CrossBit can do some optimization at the run-time according to the profile information. The profile information that CrossBit collects is the number of executing times of each BB, which can be used to find out hotspots. A hotspost is a region of contiguous code which is frequently executed. In CrossBit, a hotspot is taken as a super block, which consists of numbers of basic blocks. Because the optimization process itself is time-consuming and potentially performance degrading, it should be applied to the hot code (e.g., superblocks) instead of the cold code.

With the profile information, i.e., the number of executing times of each BB, a BB is determined to be hot if its number of times reaches a threshold, like 2000, as in the case of dynamo [13]. Once a BB becomes hot and it is not part of some superblocks, a new superblock can be constructed beginning from that BB. If the BB ends with a branch instruction, the next BB is chosen to add to the superblock according to their numbers of executing times, reversing the branch condition if necessary, like Figure 4 depicts [9]. Otherwise, it is easy to find out the next BB. This process repeats until the termination condition is met, e.g., the last instruction of the BB is an indirect jump. When the next encountered BB belongs to other superblock, the numbers of BBs reach the maximal value.

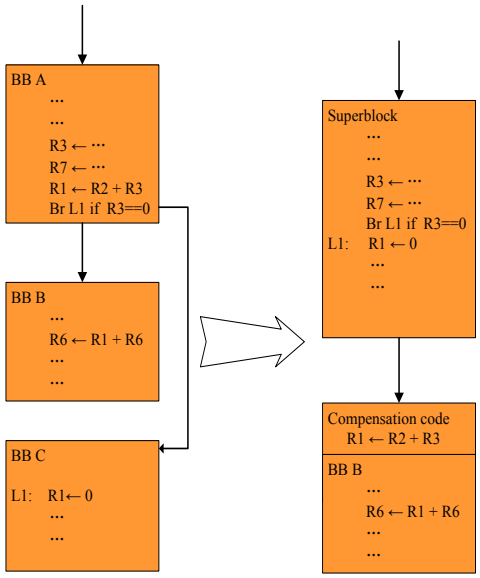


Figure 4. Build superblock based on profiling.

Another optimization tactic we adopt is modifying the direct jump instructions and indirect jump instructions. As direct jump instructions are executed frequently (nearly one in seven instructions is the direct jump), the Execution Engine has to look up into the hash table to find out the next BB after the execution of each direct jump, which leads to a bottleneck in the performance of CrossBit. A simple solution to this problem is to modify the instructions to jump directly to the next basic block, for there is only one target address for each jump. Instead of transferring the control to the Execution Engine, the modi-

 ISBN: 978-1-61208-106-9

fied instruction simply jumps to the translated instructions corresponding to the source machine instructions. This mechanism is called BB chaining and avoids significant overhead associated with returning to the Execution Engine after every instruction executed.

One of the major sources of overhead in CrossBit stems from handling indirect jumps. Experimental evidence shows that the effect of methods that handle indirect jumps depends on the features of the target architecture such as addressing modes, branch predictors, cache sizes and the ability to preserve architecture state efficiently. To tackle indirect jumps in CrossBit efficiently, we take two methods. One uses hash table to keep the target address of an indirect jump. The other mechanism is taken when some indirect jumps have only a few exits for most of the time. It uses indirect jump inlining, and the TPC of the next basic block can be obtained by comparing the jump target address to the inlined target one.

V. Other Issues

In this section, we discuss the other main issues relevant to our approach to DBT.

A. Self-Modifying Code

Self-modifying code which refers to those programs modify parts of their source code during the translation processes, though it is uncommon. When this happens, translated target code stored in the code cache would no longer correspond to the modified source code. The mechanism we adopt to handle this problem in the CrossBit is setting the original source code region write-protected. That is, any attempt to write a page in the protection area will be trapped and the delivery of a signal to the CrossBit. This can be done via a system call made by the *Execution Engine*. Consequently, the translated code will be discarded and the CrossBit invokes retranslation.

B. Address Space

Address space management is an very important aspect of DBT. In the development of CrossBit, we don't allow specific code to be placed on different regions of the memory space, like Figure 5 depicts. Because CrossBit is a process VM, it views memory as a logical address space supported by the target machine's OS. Where regions of the source memory address space could map onto regions of the target memory address space.

VI. Performance

In this section, we present preliminary results of CrossBit. Of course, no single benchmark characterizes the performance of a system, we adopt the most common method of testing, and that is running the SPECint2000 test benchmarks. Rather than giving the performance results of all the front ends and back ends we have developed, we choose two typical pairs, MIPS–*x*86 and *x*86–Power, and make an evaluation of them. The experiment of MIPS–*x*86 was taken on Intel® Pentium® 4 CPU 2.0GHz with

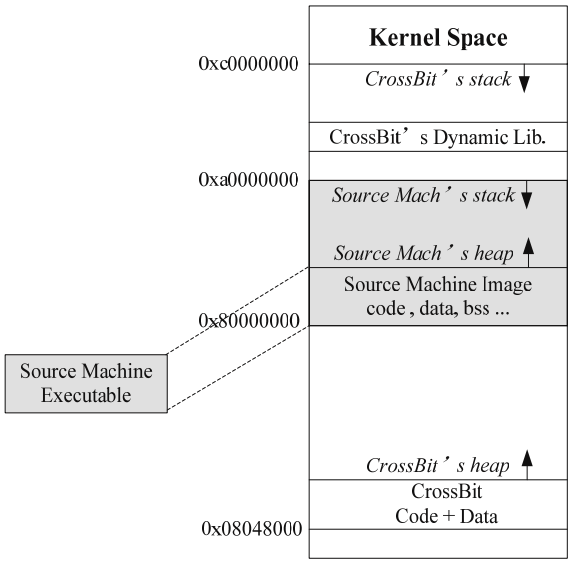
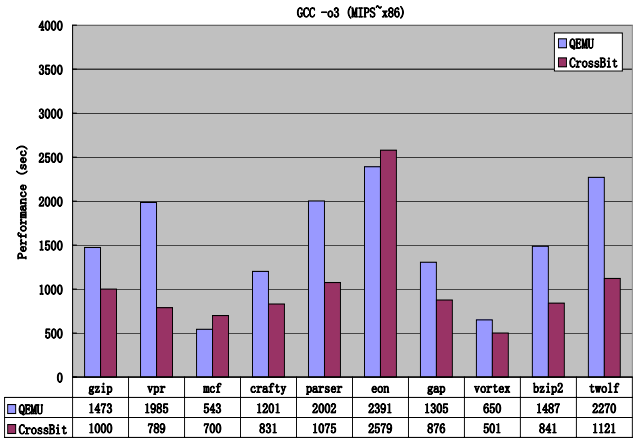


Figure 5. CrossBit runtime address space layout.

1.5GB PC3700 DDR SDRAM Memory, while the experiment of *x*86–POWER was performed on POWER® 6 1-core 4.2GHz CPU with 2 x 512 DRAM 667MHz Memory. The bottom of the charts is the executing time (sec) that every benchmark consumes to finish the testing.

Figure 6 and Figure 7 give the test results of SPECint2000 benchmarks for CrossBit translating MIPS-*x*86, and *x*86-POWER respectively. The rest of the benchmarks failed to run successfully which might be due to the lack of complete support for all Linux system calls. We still deal with these issues now. Meanwhile, in order to evaluate the performance of CrossBit, we have chosen QEMU as a reference for comparison. QEMU [12] is a multi-sources and multi-targets DBT and also using intermediate instructions. QEMU got its reputation for multi-function, not performance. QEMU itself didn't take much optimization measures.

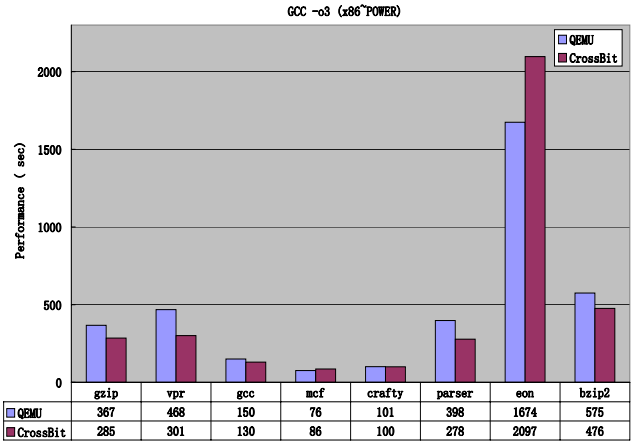


	gzip	vpr	mcf	crafty	parser	eon	gap	vortex	bzip2	twolf
QEMU	1473	1985	543	1201	2002	2391	1305	650	1487	2270
CrossBit	1000	789	700	831	1075	2579	876	501	841	1121

Figure 6. Performance comparison of the CrossBit with QEMU. The bars represent the time (sec) consumed by them from translating MIPS-*x*86 (shorter is better).

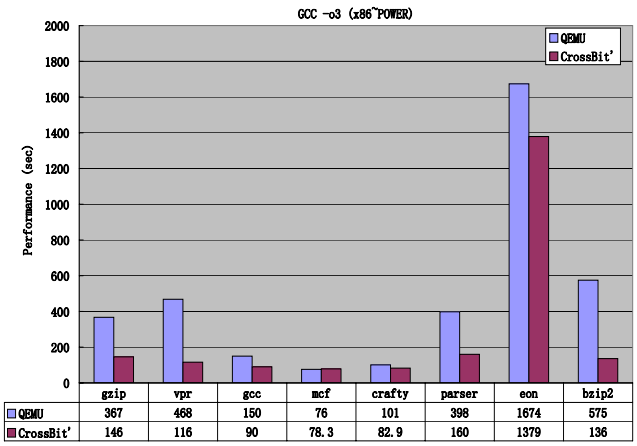
As the figures shown in Figure 6 and Figure 7, we can

 ISBN: 978-1-61208-106-9



	gzip	vpr	gcc	mcf	crafty	parser	eon	bzip2
QEMU	367	468	150	76	101	398	1674	575
CrossBit	285	301	130	86	100	278	2097	476

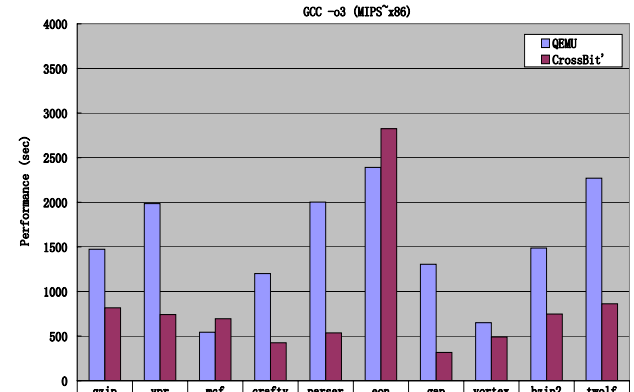
Figure 7. Performance comparison of the CrossBit with QEMU. The bars represent the time (sec) consumed by them from translating *x*86-POWER (shorter is better).



	gzip	vpr	gcc	mcf	crafty	parser	eon	bzip2
QEMU	367	468	150	76	101	398	1674	575
CrossBit'	146	116	90	78.3	82.9	160	1379	136

Figure 9. Performance comparison of the CrossBit after optimization (called CrossBit') with QEMU. The bars represent the time (sec) consumed by them from translating *x*86-POWER (shorter is better).

see that performance of CrossBit consistently outperforms than QEMU for nearly all these benchmarks. When compared with the QEMU, CrossBit performs better, since QEMU is not famous for its performance. We continue to optimization issue. Poor performance is due to code swelling. Take an example where CrossBit translates *x*86 ISA to POWER ISA, *x*86 is CISC ISA, with variable-length instructions. During the translation, from source *x*86 instructions to VInst, and from VInst to target POWER instructions, the code swelling is very big, even dozens of times. After the optimization to VInst blocks is performanced, a striking result shows that the performance of optimized code efficiency is more than 1 time faster than the natively translated code (as seen in Figure 8 and Figure 9). As CrossBit has introduced an intermediate layer to support multi-sources and multi-targets, this result is acceptable.



	gzip	vpr	mcf	crafty	parser	eon	gap	vortex	bzip2	twolf
QEMU	1473	1985	543	1201	2002	2391	1305	650	1487	2270
CrossBit'	817	741	695	425	536	2825	317	490	747	862

Figure 8. Performance comparison of the CrossBit after optimization (called CrossBit') with QEMU. The bars represent the time (sec) consumed by them from translating MIPS-*x*86 (shorter is better).

Meanwhile, we study the relationship between performance improvement and profiling overhead in Figure 10 and Figure 11. Profiling could enhance the performance of CrossBit, and also brings the extra overhead to the system. The final optimization result is obtained by the updating performance minus the overhead. Sometimes we should balance the depth of optimization and the performance of the system. From these figures, we can see that the profiling process only increases part of overheads for the CrossBit, but it does improve the quality of translated code and the execution efficiency of the whole system.

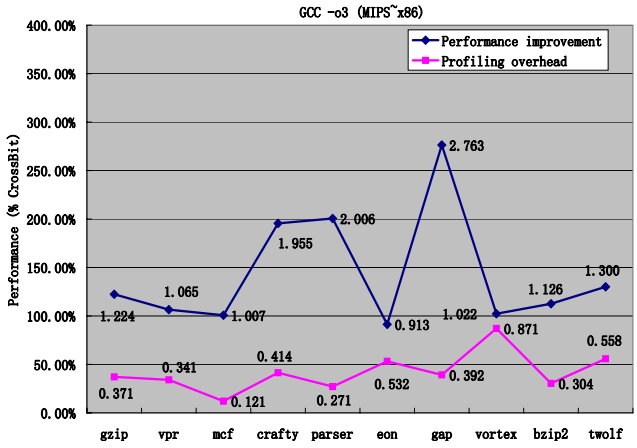


Figure 10. The relationship between performance improvement and profiling overhead of CrossBit compared with the natively translated code from MIPS-*x*86.

VII. Conclusion and Future Work

Our initial primary goal is to build a DBT platform that runs the same applications on different architectures with reasonable performance. The foundation of application migration is that instruction set should be translated correctly first. So far, we have implemented several front ends and back ends. The front ends included MIPS, *x*86 and SPARC while the back ends included *x*86, POWER and SPARC. Right now, we are focusing on improving the performance of CrossBit, so as to rival the other popular binary translators.

Our system is yet still in the absence of exceptions (traps and interrupts), and some of benchmarks fail to run

 ISBN: 978-1-61208-106-9

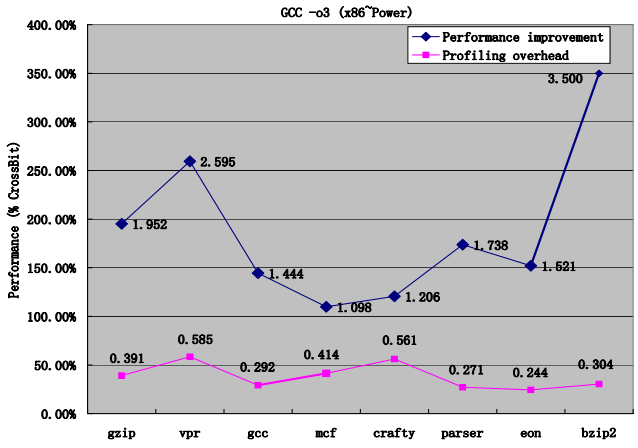


Figure 11. The relationship between performance improvement and profiling overhead of CrossBit compared with the natively translated code from *x*86-POWER.

correctly. In the future, we wish these problems would be solved.

VIII. ACKNOWLEDGMENTS

We would like to thank Yue Xu and Wei Fan for numerous discussions and for their helpful comments on how to improve the paper. Our research is supported by the National Natural Science Foundation of China (Grant No.60773093, 60970107, 60970108), the Science and Technology Commission of Shanghai Municipality (09510701600).

REFERENCES

[1] Anton Chernoff and Ray Hookway, "Running 32-Bit *x*86 Applications on Alpha NT," Proc. USENIX Windows NT Workshop, Usenix Association, Seattle, Washington, August 1997, pp. 17-23.

[2] Bala Vasanth, Duesterwald Evelyn, and Banerjia Sanjeev, "Transparent dynamic optimization: The design and implementation of Dynamo," Hewlett-Packard Laboratories Technical Report HPL-1999-78, June 1999.

[3] Wen Ke Chen, Sorin Lerner, Ronnie Chaiken, and David Gillies, "Mojo: A dynamic optimization system," ACM Workshop on Feedback-Directed and Dynamic Optimization, December 2000, pp. 81-90.

[4] Cifuentes Cifuentes and Van Emmerik Mike, "UQBT: Adaptable binary translation at low cost," Computer, vol.33, March 2000, pp. 60-66.

[5] James Dehnert, Brian Grant, John Banning, Richard Johnson, Thomas Kistler, Alexander Klaiber, and Jim Mattson, "The Transmeta Code MorphingTM Software: using speculation, recovery, and adaptive retranslation to address real-life challenges," International Symposium on Code Generation and Optimization, March 2003, pp. 15-24.

[6] Erik Altman, David Kaeli, and Yaron Sheffer, "Welcome to the Opportunities of Binary Translation," IEEE Computer, vol. 33, March 2000, pp. 40-45.

[7] Kemal Ebcioglu, Erik Altman, Michael Gschwind and Sumedh Sathaye, "Dynamic Binary Translation and Optimization," IEEE Trans, on Computers, vol. 50, June 2001, pp. 529-548.

[8] Yuncheng Bao, Haibing Guan, Jun Li and Alei Liang, "Mobilizing Native machine Code via Dynamic Binary Translation," Proceedings of the 3rd International Workshop on Software Development Methodologies for Distributed Systems, Shanghai, China, 2006, pp. 73-78.

[9] James Smith and Ravi Nair, Virtual Machines: Versatile Platforms for Systems and Processes, Morgan Kaufmann, 2005.

[10] Nicholas Nethercote and Julian Seward, "Valgrind: A program supervision framework," Electronic Notes in Theoretical Computer Science, vol.89(2), 2003, pp. 89-100.

[11] Raymond Hookway and Mark Herdeg, "Digital fx!32: combining emulation and binary translation," Digital Tech.J, vol.9(1), 1997, pp. 3-12.

[12] Fabrice Bellard, "QEMU: a Fast and Portable Dynamic Translator," Proceedings of the USENIX Annual Technical Conference, 2005, pp. 41-46.

[13] Vasanth Bala, Evelyn Duesterwald, and Sanjeev Banerjia, "Dynamo: A Transparent Dynamic Optimization System," Conf. on Programming Language Design and Implementation (PLDI), June 2000, pp. 1-12.

[14] Kevin Scott and Jack Davidson, "Strata: A software dynamic translation infrastructure," Technical Report, UMI Order Number: CS-2001-17, University of Virginia.

[15] Vikram Adve, Chris Lattner, Michael Brukman, Anand Shukla, and Brian Gaeke, "LLVA: A Low-level Virtual Instruction Set Architecture," Proceedings of the 36th annual ACM/IEEE international symposium on Microarchitecture (MICRO-36), San Diego, California, December 2003, pp. 201-216.

[16] Leonid Baraz, Tevi Devor, Orna Etzion, Shalom Goldenberg, Alex Skaletsky, Yun Wang, and Yigal Zemach, "IA-32 execution layer: a two-phase dynamic translator designed to support IA-32 applications on Itanium®-based systems," Proceedings of the 36th annual ACM/IEEE international symposium on Microarchitecture (MICRO-36), San Diego, California, December 2003, pp. 191-201.

[17] Mark Probest, "Dynamic binary translation," http://www.complang.tuwien.ac.at/schani/, May, 2009.

[18] Keith Adams and Ole Agesen, "A Comparison of Software and Hardware Techniques for *x*86 Virtualization," Proceedings of the 12th international conference on Architectural support for programming languages and operating systems, October 21-25, 2006, pp. 2-13.

[19] IBM, "Google and IBM Announced University Initiative to Address Internet-Scale Computing Challenges," October 8, 2007, http://www-03.ibm.com/press/us/en/pressrelease/22414.wss.

 ISBN: 978-1-61208-106-9

ViMo (Virtualization for Mobile) : A Virtual Machine Monitor Supporting Full Virtualization For ARM Mobile Systems

Soo-Cheol Oh, KangHo Kim, KwangWon Koh, and Chang-Won Ahn
Electronics and Telecommunications Research Institute
Daejeon, South Korea
{ponylife, khk, Kwangwon.koh, ahn}@etri.re.kr

***Abstract*—[*]This paper proposes ViMo that is a micro virtual machine monitor for ARM mobile systems, which enables to run multiple OSes on single mobile system at the same time. ViMo does not require any modification or compilation of OS, so that time and cost developing a virtualized mobile system can be reduced. Isolation of each virtual machine is another major feature of ViMo and it prevents the other virtual machines from the malfunctions of contaminated virtual machine.**

Keywords-Full Virtualization; Virtual Machine Monitor; ARM; Mobile System; Isolation;

I. INTRODUCTION

Virtualization technology of server systems has been used widely due to advantages of increased resource utilization, power saving, space saving for servers and others. The virtualization technology is about to be used in mobile and embedded systems. Mobile system virtualization has advantages of security, reduction of mobile phone BOM (Bill Of Materials) and legacy software utilization.

The virtualization technologies can be categorized into full virtualization [1] and paravirtualization [1]. In the full virtualization, the instruction modifying system status is detected in runtime and emulated by a virtual machine monitor. This technology has the advantage that it is not necessary to modify source code of a guest OS. But it has the overhead of scanning and emulating the source code and this makes system performance lower. For the paravirtualization, the source code of the guest OS is scanned offline and the instruction modifying the system status is replaced with hypercall to the virtual machine monitor or a sequence of codes having same semantic. The advantage of the paravirtualization is higher performance than the full virtualization. But it has the problem that the source code of the guest OS must be modified offline by humans or tools.

In the server systems, the full virtualization has been used popularly because server system CPUs with 3GHz frequency and multi-core provide enough performance. Also hardware virtualization technologies such as Intel-VT [2] and AMD-V [3] accelerate the full virtualization. Representative systems supporting the full virtualization are VMware [4], Parallels [5], Virtualbox of Oracle [6] and KVM [7]. Representative paravirtualized virtual machine monitor is Xen [8].

A main technology for mobile system virtualization is the paravirtualization. The largest obstacle against the mobile system virtualization is weaker performance than the server systems. The latest CPU of the server systems has 3 GHz frequency and multi-core and this provides enough performance with the virtualization environment. However, the mobile systems have generally 500 ~ 800Mhz CPU with single core and doesn't provide enough performance with the virtualization systems. This limitation makes main virtualization trend of the mobile systems as the paravirtualization. But, the latest mobile systems including iPhone 4G of Apple and Galaxy S of Samsung are based on 1GHz ARM processor. Also the smart phones including HTC Desire, HTC HD2 with 1GHz Qualcomm SnapDragon are being delivered to the users. Thus the performance issue that is the main obstacle of the mobile system virtualization is being solved. Also, it is scheduled that new ARM core will support a hardware virtualization, and mobile systems with multi core will appear soon. Consequently, fundamentals for the full virtualization in the mobile systems will be concrete.

The paravirtualized mobile systems are MVP of VMware [9], VLX of VirtualLogix [10] and XenARM [11]. The full virtualizated mobile systems are QEMU [12].

This paper presents ViMo (Virtualization for Mobile) that is a virtual machine monitor based on the full virtualization for the mobile systems. ViMo scans binary code of OS in runtime and emulates critical instructions. Also ViMo allocates physical memory space to each virtual machine and provides memory isolation among the virtual machines.

II. VIMO ARCHITECTURE

ViMo is the virtual machine monitor supporting the full virtualization for ARM-based mobile systems and the structure of ViMo is shown in Fig. 1. ViMo works on system hardware and multiple OSes are mounted on ViMo. ViMo creates one virtual machine per each OS and the OS runs on the corresponding virtual machine.

In typical systems, OS and application are located in supervisor (SVC) and user (USR) mode of ARM CPU, respectively. However, the OS in the ARM's SVC mode must be moved to the ARM's USR mode because ViMo is inserted into the ARM's SVC mode. In the virtualized systems based on ViMo, the USR mode of ARM, in which the OS is executed, is called logically as Virtual SVC

[*] This work was supported by the MKE [KI002088, Development of Virtualization Technology based on Open Source Software]

 ISBN: 978-1-61208-106-9

(VSVC) and the USR mode, in which applications are executed, is called Virtual USR (VUSR). The four modes used in ViMo are summarized as Table I.

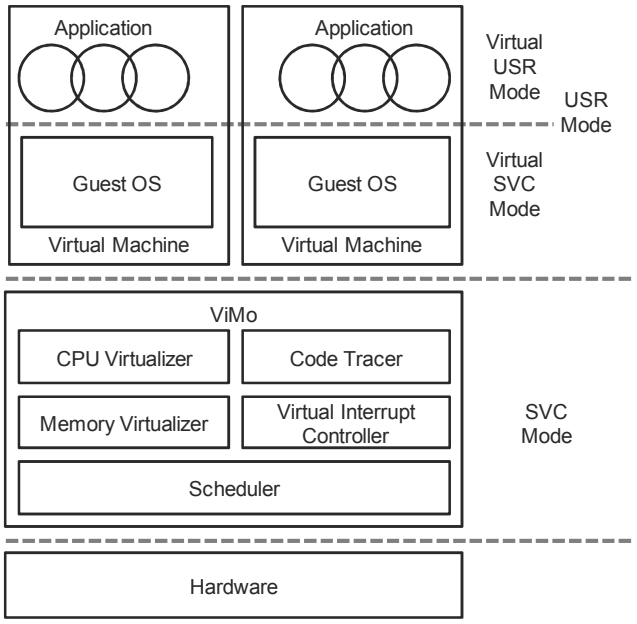


Figure 1. ViMo Architecture

TABLE I. CPU MODE OF VIMO

Mode	Descriptions
ASVC	(ARM SVC) SVC mode of ARM Hardware
AUSR	(ARM USR) USR mode of ARM Hardware
VSVC	Mode in that OS is executed in the ViMo-based virtualization system
VUSR	Mode in that applications are executed in the ViMo-based virtualization system

ViMo has mainly five components that are code tracer, CPU virtualizer, memory virtualizer, interrupt controller virtualizer and scheduler. The code tracer detects critical instructions on the virtual machines in runtime and the CPU virtualizer emulates the detected critical instructions. The memory virtualizer allocates memory space to the virtual machines and isolates the virtual machine from other virtual macines. The interrupt controller virtualizer builds a virtual interrupt controller for each virtual machine and processes interrupt controller accesses from the virtual machine. The scheduler switches the virtual machines periodically.

III. MODE TRANSITIONS

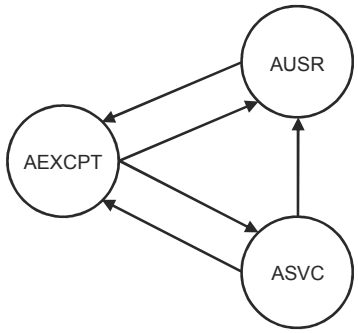
Typical systems not using ViMo have three modes. Two modes of them are ASVC and AUSR explained in table I and other mode is AEXCPT handling exceptions. Exception modes including interrupt (irq), fast interrupt (fiq), prefetch abort, data abort and SWI (software interrupt) can be explained as AEXCPT. As shown in Fig. 2-(a), the transition from ASVC running OS' kernel to AUSR running applications is performed directly. If AUSR wants to use kernel service using system calls, AUSR must be transited to ASVC through AEXCPT. Also ASVC or AUSR are transited to AEXCPT if exceptions happen, and after exception serving is completed, AEXCPT is changed directly to ASVC or AUSR.

In ViMo, VSVC, VUSR and VEXCPT are added instead AUSR as shown in Fig. 2-(b). The guest OS is executed in the virtual modes including VSVC, VUSR and VEXCPT which are located in AUSR. ViMo is executed in ASVC and AEXCPT.

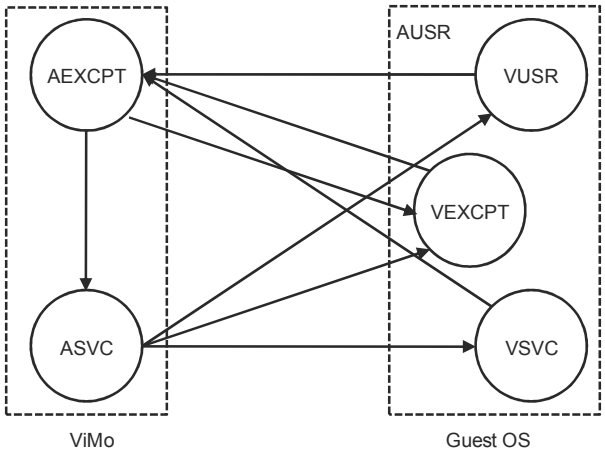
The transitions among VSVC, VUSR and VEXCPT cannot be performed directly because these modes are located in AUSR. Thus, these mode transitions must be made by ASVC and AEXCPT.

The mode is transited directly to AEXCPT if an exception happens when CPU is in VSVC, VUSR or VEXCPT. In this situation, there are two processing options. If the exception is for the guest OS, this exception is passed to the guest OS and mode is transited from AEXCPT to VEXCPT. If the exception is for ViMo, the mode is transited to ASVC.

In ASVC, the transition to AEXCEPT is disabled. ViMo processes virtual machine management jobs including guest OS scheduling, memory management and critical instruction emulation, and these jobs have atomic characteristic that they must be processed without break. To preserve this characteristic, exception generation in ASVC is prohibited.



(a) System not using ViMo



(b) System using ViMo

Figure 2. Mode Transition of ViMo

IV. CPU VIRTUALIZATION

This section explains CPU virtualization of ViMo. In ViMo, the guest OS is executed in VSVC and VUSR that belong to AUSR. VUSR doesn't make problems because both of VUSR and AUSR are user mode. However, the guest OS kernel running in VSVC doesn't generate the same result as ASVC because VSVC is located in AUSR. This problem must be solved by ViMo.

There are 148 instructions in ARM architecture v6 [13]. In these instructions, instructions related to the virtualization can be categorized as privileged, sensitive and critical instruction. The privileged instruction must be executed only in the privileged mode (ASVC). If this instruction is executed in AUSR, an exception is generated and control is taken by the privileged mode. The sensitive instruction generates different results when the instruction is executed in ASVC or AUSR. The critical instruction is the sensitive instruction but not the privileged instruction.

To virtualize a system using the full virtualization, the instructions that modify the system status must be executed under control of ViMo or be replaced with a sequence of instruction having same semantic. The instructions modifying system status are the privileged and the critical instructions. If the privileged instruction is executed in VSVC (AUSR mode), an exception is generated and control is taken by ASVC running ViMo. Thus, detection of the privileged instruction is performed automatically by ARM CPU.

If the critical instruction is executed in VSVC (AUSR mode), CPU makes no error and generates the result for AUSR mode that is different from intended result for ASVC mode. Thus, this is one of the most important problems to be solved in ARM CPU virtualization.

Table II shows instruction categorization for ARM CPU. There are 6 privileged instructions in ARM architecture with 148 instructions. MCR and MRC that belong to the privileged instructions have over 100 operand combinations. In detail, MCR and MRC have both characteristics of the privileged and sensitive instructions. Some operand combinations have the privileged instruction characteristic and other operand combinations have the sensitive instruction characteristic. Consequently, each operand combination of MCR and MRC must be handled as single instruction.

TABLE II. PRIVILEGED AND CRITICAL INSTRUCTIONS

Instruction	# of Inst.	Descriptions
Privileged instruction	6	CDP, LDC, MCR, MCRR, MRC, MRRC
Critical instruction-1	14	CPS, LDM(2), LDM(3), LDRBT, LDRT, MRS, MSR, RFE, SETEND, SRS, STC, STM(2), STRBT, STRT
Critical instruction-2	13	ADC, ADD, AND BIC, EOR, MOV, MVM, ORR, RSB, RSC, SBC, SUB, LDR

The critical instruction consists of the critical instruction-1 and the critical instruction-2. The critical instruction-2 can be critical instruction upon operands. Generally, the critical instruction-2 is not the privileged or the critical instruction. However, if the critical instruction-2 has S-bit option and PC as the destination operand, this instruction becomes the critical instruction. This operand combination tells that SPSR (Saved Processor Status Register) of current mode is copied to CPSR (Current PSR). However, if this operand combination is executed in AUSR, the instruction result is unpredictable because AUSR mode doesn't have SPSR.

Fig. 3 shows structure of the CPU virtualization. Basic Block Identifier (BBI) scans a binary image in runtime and identifies basic block. A basic block is the code block with single entry and single exit point. The exit point may be branch or the critical instruction. Suppose that a basic block begins at address 10 and a critical instruction is at address 20.

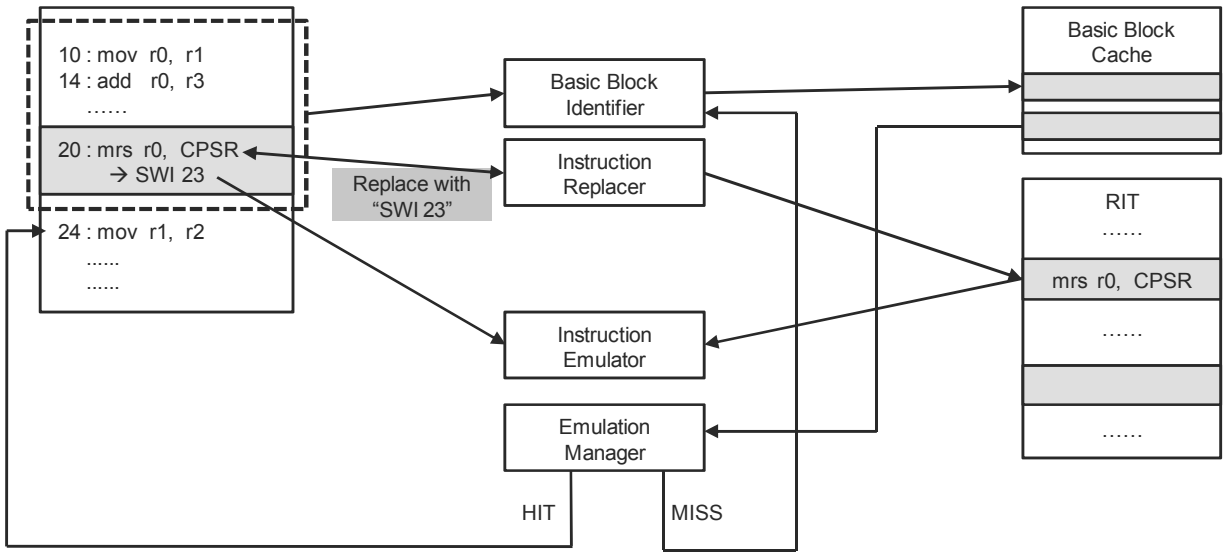


Figure 3. CPU Virtualization Architecture

BBI begins to scan the code at address 10. When BBI meets the critical instruction at address 20, it makes a basic block and stores this block at basic block cache. The basic block cache stores the basic block that is already scanned. When the basic block stored at the basic block cache is executed again, it is not necessary to scan the block.

Then BBI calls instruction replacer. The instruction replacer stores the critical instruction at Replacement Instruction Table (RIT) and replaces the critical instruction with single SWI instruction having index to RIT. For example, the critical instruction is replaced with "swi 23" if the instruction is stored at 23th index of RIT.

The instruction replacer sets PC register as the entry point (address 10 in Fig. 3) of the basic block and executes the basic block. Then codes at address 10 to 1C are executed and the SWI instruction at address 20 generates an exception that delivers control to ViMo located in ASVC. This sequence was already explained in Fig. 2-(b). ViMo calls instruction emulator. The instruction emulator retrieves index to the RIT from the SWI instruction, looks up the RIT and loads the original instruction (mrs r0, CPSR) that is replaced with the SWI instruction. Then, the original instruction is emulated by the instruction emulator. After completion of the instruction emulation, ViMo calls BBI and continues to find next basic block beginning at address 24. BBI skips to find the basic block if the basic block is cached in the basic block cache. The sequence explained above is repeated until the corresponding virtual machine is shutdown.

V. MEMORY VIRTUALIZATION

ViMo virtualizes main memory on system and provides memory space with each virtual machine. ViMo also isolates virtual machine from each other by preventing accesses to memory of other virtual machines without permission of ViMo.

The memory virtualization architecture of ViMo is shown in Fig. 4. For example, suppose that a system has a main memory with 128MB located at address 0x50000000. In ViMo, address space is categorized as three address spaces that are virtual address, physical address and machine address. The virtual address space is used by the guest OS and the physical address space is recognized as real physical address by the guest OS. The machine address space is maintained by ViMo.

ViMo allocates a machine address space to each virtual machine and this memory space is contiguous physically. This address space is not changed until the corresponding virtual machine is shutdown.

Suppose that ViMo allocates 32MB region of the machine address space at 0x51000000 ~ 0x52FFFFFF to the guest OS 0. In this situation, ViMo provides the illusion, that this address is located at 0x50000000 ~ 0x51FFFFFF, with the guest OS 0. The guest OS 0 maps the physical address to virtual address and uses this address space.

The same memory allocation is applied to the guest OS 1. ViMo allocates the machine address space at 0x53000000 ~ 0x54FFFFFF to the guest OS 1 and the guest OS 1 thinks that it uses the physical address space at 0x50000000 ~ 0x51FFFFFF.

ViMo uses shadow page table (SPT) [14] for the memory virtualization. The guest OS has its own guest page table maintaining the memory mapping from the virtual address to the physical address. ViMo knows memory mapping from the physical address of each virtual machine to the machine address. Thus, ViMo can create the memory mapping from the virtual address of each virtual machine to the machine address and this memory mapping is maintained by SPT.

The guest page table is modified continuously by the guest OS. ViMo should detect modification of the guest page table and update SPT dynamically to reflect this modification. To solve this problem, ViMo modifies an attribute of the memory region that stores the guest page table from read-write to read-only. Memory write to the guest page table by the guest OS generates an exception and control is taken by ViMo. Thus ViMo can capture the page table write and maintain SPT.

VI. INTERRUPT CONTROLLER VIRTUALIZATION

Fig. 5 shows virtualization architecture of interrupt controller in ViMo. ViMo builds a Virtual Interrupt Controller (VIC) based on HW Interrupt Controller (HWIC) and provides it to the guest OS. ViMo makes one VIC per

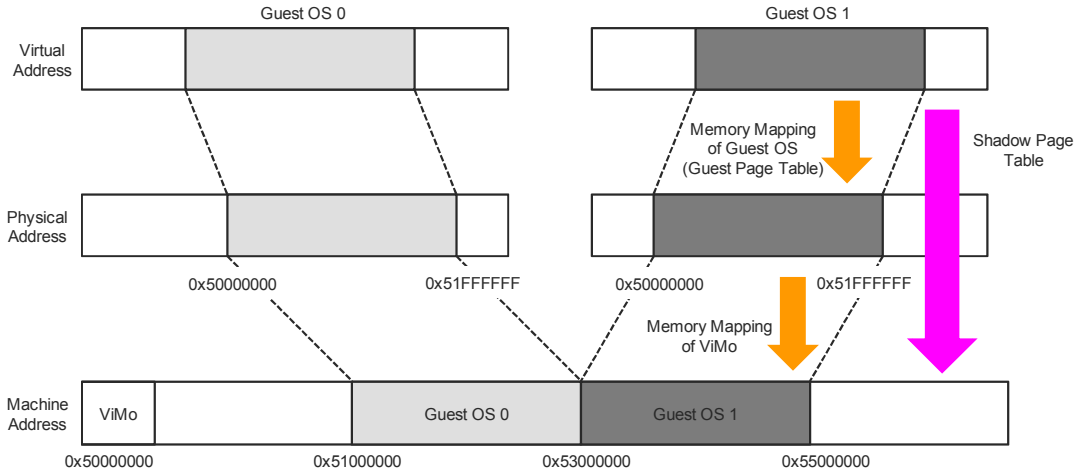


Figure 4. Memory Virtualization Architecture

each guest OS. Thus, the guest OS uses VIC instead of HWIC.

The guest OS controls HWIC by accessing registers in HWIC that are mapped to address space of CPU. Thus it is necessary to virtualize the registers for HWIC virtualization. VIC has same register configuration as HWIC. The access controller of VIC handles access from the guest OS and maintains consistency between VIC and HWIC.

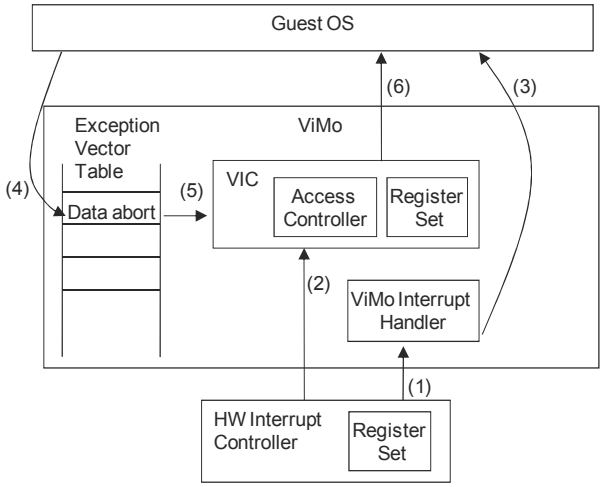


Figure 5. Interrupt Handling

If a HW interrupt is generated, ViMo interrupt handler is activated (Fig. 5-(1)) and copies interrupt information from HWIC to VIC (Fig. 5-(2)). In this situation, the ViMo interrupt handler checks interrupt mask of VIC and doesn't copy the interrupt that is masked on VIC. Then, ViMo calls interrupt handler of the guest OS (Fig. 5-(3)).

The interrupt handler of the guest OS tries to access HWIC for getting what kind of interrupt is generated. When the guest OS accesses HWIC, ViMo intercepts this access (Fig. 5-(4)). HWIC has some internal registers that are mapped to address space of CPU and the guest OS tries to access HWIC using this registers. ViMo modifies memory mapping between the guest OS and HWIC, so that the guest OS has no memory mapping for HWIC. In this case, the access to HWIC by the guest OS generates a data abort exception. If the data abort exception is generated and the exception handler of ViMo confirms that this exception is for HWIC, the access controller of VIC is activated (Fig. 5-(5)). The access controller processes the access request using VIC and maintains the consistency between HWIC and the VIC. After processing the access from the guest OS, control is returned to the guest OS that calls HWIC access (Fig. 5-(6)).

VII. SCHEDULER

Scheduler of ViMo switches virtual machines periodically. The scheduler stores status of virtual machine executed during previous time quantum at main memory and loads status of next virtual machine. The used scheduling algorithm is round-robin.

VIII. IMPLEMENTATION

ViMo proposed in the paper was implemented on an ARM11-6410SYS board developed by Huins[15] that is a embedded development board based on ARM11. The CPU is Samsung S3C6410 using ARM1176JZF-S core and frequency is 533MHz. The board has 128MB DDR SDRAM, 128MB NAND flash and 1MB NOR flash. The Board also has 4.3 inch wide color TFT LCD with 480x272 resolution. We executed Linux with 2.6.21 kernel and uC/OS-II as the guest OS. Binary code size of ViMo is 34KB that is very small.

IX. EXPERIMENTAL RESULTS

The experiments were performed with three cases that are RawLinux, ViMo-Single and ViMo-Dual. RawLinux is a case not using ViMo. ViMo-Single and ViMo-Dual are ViMo systems with single guest OS and two guest OSes, respectively.

A. *DhryStone*

We measured performance of ViMo using DhryStone [16]. DhryStone is the benchmark measuring the CPU performance developed by Dr. Reinhold P. Weicker. The number of run in DhryStone is 10,000,000.

RawLinux, ViMo-Single and ViMo-Dual show 432, 271, and 150 VAX MIPS, respectively. We know that ViMo-Single is 37% slower than RawLinux and this overhead is from ViMo. ViMo-Dual is 44% slower than ViMo-Single because ViMo-Dual has two guest OSes.

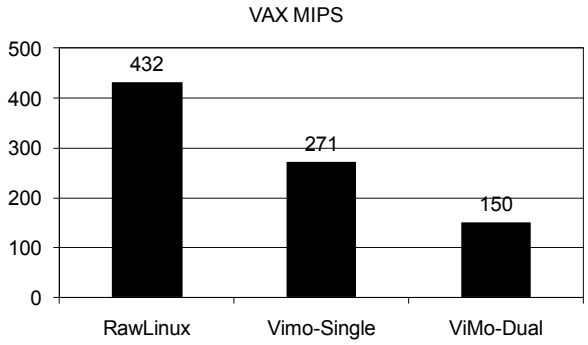


Figure 6. DhryStone Benchmark

Figure 7. Moive Play Capture

B. Play Movie

This experiment is to play a movie in ViMo-Single. The profile of the movie is 480x272 resolution and 30 frame/s. By using ViMo, the total play time of ViMo-Single increases compared to RawLinux. The play time of the movie file is 30 seconds but the play time in ViMo-Single increases to 40 seconds. Thus, ViMo produces 33% overhead.

C. Virtual Machine Isolation

ViMo provides the isolation of the virtual machines and the experiment shows that a malfunction from one virtual machine is not propagated to other virtual machines. In ViMo-Dual, movie play is executed in the guest OS 0 and a kernel module generating kernel panic is loaded in the guest OS 1. The kernel modules generates the kernel panic by writing memory region that doesn't belong to the guest OS 1. Fig. 8 shows that the guest OS 1 is stopped by the kernel panic. However, the movie play in the guest OS 0 still works without errors.

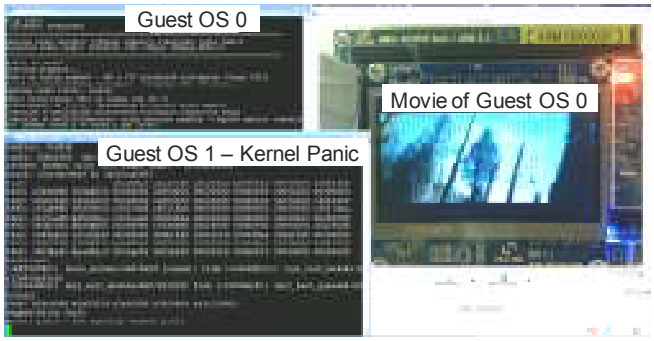


Figure 8. Virtual Machine Isolation

D. Performance Analysis

The experimental results of section A and B shows that ViMo is not acceptable for using real mobile systems because of about 33 ~ 37% overhead. The implementation presented in this paper is in its initial state and ViMo has many performance improvement points.

We estimate that the most time consuming part of ViMo is the critical instruction detection, the instruction emulation and related context switches between the guest OS and ViMo. ViMo generates many context switches that are from the critical instruction emulation, the basic block identification, the virtual interrupt controller accessing and the guest OS switching. These frequent context switches between the guest OS and ViMo make cache of CPU flushed and this makes the performance degradation.

We are improving the critical instruction detection and the instruction emulation algorithm to minimize the context switches between the guest OS and ViMo. We are also improving other components of ViMo including the memory virtualization and the interrupt virtualization. Futhermore, we are developing ViMo for ARM Cortex-A8.

Our aim is that the overhead becomes below 10% through these improvement works.

X. Conclusions and Future Works

This paper proposed ViMo that is the virtual machine monitor using the full virtualization for mobile systems based on ARM architecture. ViMo provides the virtualization for CPU, memory and interrupt controller. The binary image is scanned in runtime and the critical instructions are replaced with SWI to ViMo. The replaced instructions are emulated in runtime. Also ViMo presents the memory virtualization and each virtual machine has its own memory space provided by ViMo. A virtual machine cannot access the memory space of other virtual machines without permission of ViMo and no fault of one virtual machine is propagated to other virtual machines. VIC virtualize the HW interrupt controller and each virtual machine has its own VIC.

According to the experimental results, ViMo has 33 ~ 37% overhead. We are improving all components including the critical instruction detection, the instruction emulation and the memory virtualization algorithm. Our aim is that the overhead becomes below 10% through these improvement works. Additionally, we are under designing I/O virtualization

References

[1] "Understanding Full Virtualization, Paravirtualization, and Hardware Assist", White Paper, pp.4-5, VMware, 2007.

[2] Gil Neiger, Amy Santoni, Felix Leung, Dion Rodgers, Rich Uhlig, "Intel Virtualization Technoloyg: Hardware Support for Efficient Processor Virtualization", Intel Technology Journal, Vol. 10, Issue 03, pp. 170-175, Auguest, 2006.

[3] "AMD Virtualization Technoloyg", http://sites.amd.com/us/business/it-solutions/virtualization/Pages/amd-v.aspx, AMD, 2010.

[4] "Building the Virtualized Enterprise with VMware Infrastructure", White Paper, pp. 4-5, VMware, 2008.

[5] http://www.parallels.com, 2010.

[6] Virtualbox "Virtualbox Architecture", http://www.virtualbox.org/wiki/VirtualBox_architecture, Oracle, 2010

[7] AMIT SHAH, "Kernel-based Virtualization with KVM", Linux Magazine, Issue 86, p.37-39, Jan, 2008.

[8] Paul Barham, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris, Alex Ho, Rolf Neugebauer, Ian Pratt, Andrew Warfield, "Xen and the Art of Virtualization", Proceedings of the nineteenthACM Symposium on Operating Systems Principles, pp. 164-177, 2003.

[9] "VMware MVP", http://www.vmware.com/products/mobile/index.html, VMware, 2010.

[10] "Meeting the Challenges of Connected Device Design", White Paper, VirtualLogix, pp. 8-10, 2006.

[11] Joo-Young Hwang, Sang-Bum Suh, Sung-Kwan Heo, Chan-Ju Park, "Xen on ARM: System Virtualization using Xen Hypervisor for ARM-based Secure Mobile Phones", IEEE 5th Consumer Communications and Networking Conference, pp. 257-261, 2008.

[12] Fabrice Bellard, "QEMU, a Fast and Portable Dynamic Translator", USENIX 2005 Annual Technical Conference, pp. 41-45, 2005.

[13] "ARM Architecture Reference Manual", ARM, pp. 152-435, 2005.

[14] James E.Smith, Ravi Nair, "Virtual Machines, Versatile Platforms for Systems and Processes", Morgan Kaufmann Publishers, p.399-402, 2005.

[15] http://www.huins.com, HUINS, 2010.

[16] Alan R. Weiss, "Dhrystone Benchmark, History, Analysis, Scores and Recommendations", White paper, November, pp. 1-5, 2002.

 ISBN: 978-1-61208-106-9

Probabilistic Virtual Machine Assignment

David Wilcox
CS Deptartment, BYU
Provo, USA
davidw@cs.byu.edu

Andrew McNabb
CS Deptartment, BYU
Provo, USA
a@cs.byu.edu

Kevin Seppi
CS Deptartment, BYU
Provo, USA
k@cs.byu.edu

Kelly Flanagan
CS Deptartment, BYU
Provo, USA
kelly_flanagan@byu.edu

***Abstract*—We cast the assignment of virtual machines (VMs) to physical servers as a variant of the classic bin packing problem. We then develop a model of VM load that can be used to produce assignments of VMs to servers. Using this problem formulation and model, we evaluate heuristic solutions to this problem. We evaluate the performance of these solutions in stochastic load environments. We verify the model proposed and show that it can be adapted to respond well to varying VM loads.**

***Keywords*-Energy optimization; Probabilistic Models**

I. Introduction

One of the major causes of energy inefficiency in data centers is the idle power wasted when servers run at low utilization [17]. In 2005, data centers accounted for 0.8% of all world energy consumption, costing $7.2 billion (US) [9]. Part of the problem is that most servers and desktops are in use only 5-15% of the time they are powered on, yet most x86 hardware consumes 60-90% of normal workload power even when idle [20], [4], [5].

Data center costs can be reduced by utilizing virtual machines (VMs). Using virtualization, multiple operating system instances can run on the same physical machine, exploiting hardware capabilities more fully, allowing administrators to save money on hardware and energy costs. To maximize the savings, administrators should assign as many VMs as possible to servers given performance requirements. We refer to this problem as the *Virtual Machine Assignment Problem*.

The Virtual Machine Assignment Problem (VMAP) is the problem of, given probabilistic distributions over the VM load, find an initial assignment which distributes the load on the VMs such that all have access to adequate resources and the number of servers used is minimized.

This type of problem might need to be solved at an e-commerce web site, which generally have times of lower hardware utilization. During these times of lower utilization, the web site acquires very few sales and therefore has more liberty to move VMs around. Once the day begins however, traffic will pick up and administrators will be less able to move VMs around. A good initial assignment means moving fewer VMs during peak hours of production. VMAP is not the problem of reassigning load after an initial assignment has already been made. We address this issue separately.

VMAP can be seen as a type of Bin Packing Problem, the problem of assigning a set of items into a set of bins, minimizing the number of bins in use [8]. However, VMAP is not as simple as the conventional Bin Packing Problem. VMAP is different in two important ways.

1) Each server has multiple types of constrained resources which the VMs consume. Each VM adds some amount of load to each resource type provided by the server, such as memory, disk space and CPU.
2) Loads that VMs exert on servers are probabilistic and not completely known ahead of time.

Prior research has partially addressed VMAP [15], [18]. One thing that prior research has not discussed is what happens when loads are probabilistically distributed. Specifically, we wish to investigate how load distributions can be incorporated into packing virtual machines onto servers. Prior research has not focused on this specific sub-aspect of VMAP.

This paper presents a novel way of taking expectations on loads of virtual machines. If system administrators have knowledge about the loads of virtual machines, expectations can be taken at different points in the probabilistic load distribution. The purpose of this paper is to present a way that this process can be done and to investigate some consequences of treating loads probabilistically.

We outline the paper here for reference. In Section II, we describe our background research. In Section III, we describe the model that we propose in this paper. In Section IV, we describe the assignment algorithms that we will compare. In Section V, we discuss the metrics to determine success in our results. In Section VI, we detail our experimental setup. Finally, in Section VII, we discuss the results of our experiments.

II. Background Research

Different aspects of server consolidation have been studied and modeled [16], [14], [19], [3]. Other literature has focused on decreasing power use in virtualized environments [12]. In this section we review background research in three parts. First, prior research on modeling server load will be discussed. Second, We will discuss the Bin Packing problem. Third, we will briefly describe Genetic Algorithms as this is the foundation for one solution technique.

A. Virtual Machine Assignment

The problem that we identify in this paper as VMAP takes other names in literature. Stillwell et al. [18] define ResAlloc, which is a Mixed Integer Linear Program formulation of

 ISBN: 978-1-61208-106-9

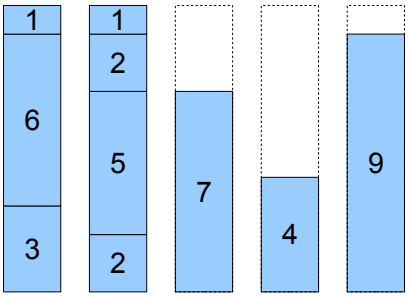


Fig. 1. An inefficient Bin Packing solution.

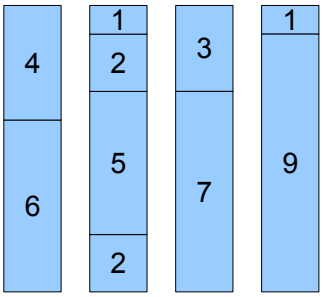


Fig. 2. An efficient Bin Packing solution.

VMAP. In their problem formulation, they consider maximizing the yield, which represents the faction of a job's achievable compute rate that is achieved, on the server with the minimum yield. They then identify different solutions and evaluate the solutions on ResAlloc.

Song et al. [15] created RAINBOW, a prototype to evaluate a multi-tiered resource scheduling scheme on a workload scenario reflecting resource demands of services in a real enterprise environment. They first define resource flowing as the process in which resources released by some VMs/services are allocated to others. Their main contributions were a multi-tiered resource scheduling scheme for a VM-based data center, a model for resource flowing using optimization theory, and a global resource flowing algorithm.

Something that has not been investigated well in prior research is the fact that virtual machines are probabilistically distributed. We investigate what happens when we know something about the probabilistic nature of loads on virtual machines. This paper will give system administrators knowledge on what they should do, given that they have prior knowledge of how their virtual machines are distributed.

B. Conventional Bin Packing

In this paper we build a model that, given a set of VMs, each with a distribution of resource utilizations, decides how VMs should be assigned to servers. We present our model as a variant of the Conventional Bin Packing Problem. The Bin Packing Problem is the problem of finding the assignment of items to bins under which the number of bins is minimized.

Definition 1. The *Bin Packing Problem* is formulated as follows. Given a finite set of n items $I = \{1, 2, \ldots, n\}$ with corresponding weights $W = \{w_1, w_2, \ldots, w_n\}$ and a set of identical bins each with capacity $\mathcal{C}$, find the minimum number of bins into which the items can be placed without exceeding the bin capacity $\mathcal{C}$ of any bin. A solution to the Bin Packing Problem is of the form $B = \{b_1, b_2, \ldots, b_m\}$, where each b_i is the set of items assigned to bin i, and is subject to the following constraints:

1) $\forall i \, \exists! \, j$ such that $i \in b_j$ (Every item belongs to one unique bin.)
2) $\forall j \sum_{n \in b_j} w_n \leq \mathcal{C}$ (The sum of the weights of items inside any bin cannot be greater than the bin capacity.)

In the Bin Packing Problem, the objective is to assign the set of items into a set of bins, minimizing the number of bins used. This idea is illustrated in Figures 1 and 2. Figure 2 shows an assignment that uses the same items as the assignment shown in Figure 1, but packs them in fewer bins.

Doing an initial placement of VMs onto servers can be seen as a type of Bin Packing Problem. In the Bin Packing Problem, a set of items are placed into bins, minimizing the number of bins. In the problem of placing VMs onto servers, VMs are assigned to servers, minimizing the number of servers, while assuring that some performance criteria is met. Because these two problems are similar in this way, we will model the problem of making an initial assignment of items to servers as a type of Bin Packing Problem.

The Bin Packing Problem has been shown to be NP Hard [2]. We solve the problem that Bin Packing is inherently intractable by using approximation algorithms to solve the problem in our experiments.

One reason why the conventional Bin Packing Problem itself cannot be used to model the VM Assignment Problem is that there is no way in the conventional Bin Packing Problem to model multiple resources on one server. For that reason, we defer to prior research and model this problem using the Multi-Capacity Bin Packing Problem [21]. We will describe this problem in more detail in Section II-C.

C. Multi-Capacity Bin Packing Problem

Definition 2. The *Multi-Capacity Bin Packing Problem* or *Vector Packing Problem* is similar to the conventional Bin Packing Problem that was given in Definition 1, but not identical. In the Multi-Capacity Bin Packing Problem, the capacity is a d-dimensional vector $\mathcal{C} = \langle C_1, C_2, \ldots, C_d \rangle$ where d is the number of resource types. The weights are redefined so that the weight of item i is a d-dimensional vector $\vec{w_i} = \langle w_{i_1}, w_{i_2}, \ldots, w_{i_d} \rangle$ [18], [10].

1) $\forall i \, \exists! \, j$ such that $i \in b_j$ (Every item has to belong to some unique bin.)
2) $\forall j \, \forall k \sum_{n_k \in b_j} w_{n_k} \leq C_k$ (The sum of all the weights for any capacity for any bin must be less than the corresponding capacity for that bin.)

Note that in the Multi-Capacity Bin Packing Problem the resources consumed by each VM accumulate, up to some maximum. Items are placed on each corner to show that

 ISBN: 978-1-61208-106-9

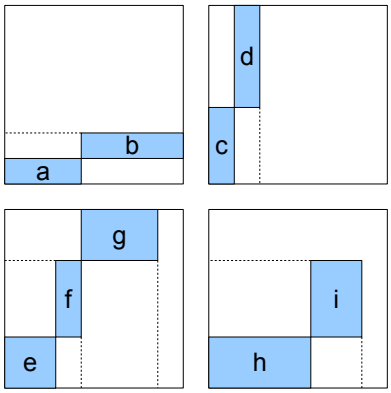


Fig. 3. An inefficient Multi-Capacity Bin Packing solution.

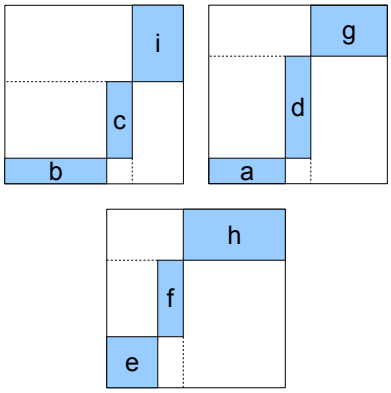


Fig. 4. An efficient Multi-Capacity Bin Packing solution.

the sum of the weights for any one type has to be less than the corresponding bin capacity. The weights on items in the same bin are additive. For all resources, the sum of all weights on items of that particular resource must be less than the corresponding resource capacity on the server.In VMAP, resources used by one VM can not be used by any VM on the same physical hardware.

As with the conventional Bin Packing Problem, the objective is to minimize the number of bins. Figures 3 and 4 illustrate this principle. Figure 4 packs the exact same items as are found in Figure 3 in three bins instead of four. This means one fewer server drawing power.

The Multi-Capacity Bin Packing Problem lends itself better to the problem of assigning VMs onto servers than the conventional Bin Packing Problem. In the conventional Bin Packing Problem, there is no way to model the multiple different resources that are available on a server, such as CPU, RAM and disk bandwidth. Because the Multi-Capacity Bin Packing Problem lends itself well to modeling the many different resources of a server, we use this problem formulation in our optimization techniques.

D. Genetic Algorithms

Bin Packing Problems have been solved with Genetic Algorithms (GAs). There is no rigorous definition for GAs [11]; they derive much of their inspiration from Darwinian biological processes. In GAs, individuals represent candidate solutions to the problem. These candidate solutions explore the solution space by undergoing processes similar to those of biological organisms. The simplest form of genetic algorithm involves three types of operators:

- **Selection–**Individuals in the population are selected for crossover with other individuals. Usually, selection is based on elitism, where the more fit individuals are selected more often than less fit individuals.
- **Crossover–**Two individuals in the population exchange subsections of their candidate solution with each other to create new offspring.
- **Mutation–**After crossover, each individual has a probability of having their candidate solution modified slightly.

III. Description of Model

As described in Section II, we model VMAP by using the Bin Packing Problem. However, there are a few differences between the Bin Packing Problem and VMAP which were identified in Section I. The model that we will use for VMAP is based on the Multi-Capacity Bin Packing Problem, as discussed in Section II-C, and it also uses probabilistic estimates of loads, which we will discuss here.

A. Probabilistic Estimates

The second way that VMAP is different from the conventional Bin Packing Problem is that the loads VMs exhibit on servers are not known completely when initial assignments are made. Even though these loads are not completely known ahead of time, probabilistic estimates can be made for loads on VMs. Therefore, we treat loads as probabilistic. There are at least two ways in which system administrators can derive probabilistic estimates for loads on VMs.

1) If the system administrators have reason to believe that VM loads can be characterized as a type of known probabilistic distribution, this problem becomes the problem of parameter estimation. Using data, it is possible to estimate the parameters of a parametric probabilistic model using known methods such at methods of moment or maximum likelihood estimation.
2) If system administrators do not know the probabilistic distribution which describes the load on VMs, a *nonparametric model* can be used. We will describe nonparametric distributions in more detail in Section VI.

We assume that the probabilistic distribution on the future load for each resource for each VM is to the algorithm ahead of time. Recall as well that in the Multi-Capacity Bin Packing Problem, weights are deterministic and known instead of probabilistic and unknown. This means that if the Multi-Capacity Bin Packing Problem is to be used as a model, some estimate or expectation of the probabilistic distribution must be given to the algorithm solving VMAP. In this section, we will explore how to make this estimate from probabilistic distributions of the load.

Recall that the *probability density function (pdf)* $f(X)$ of a random variable X describes the relative likelihood of X

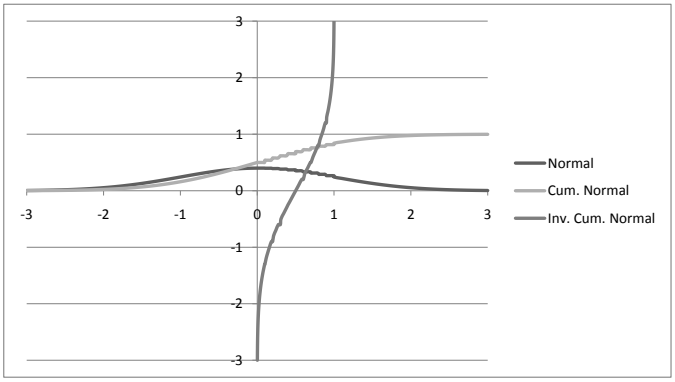


Fig. 5. The pdf, cdf and icdf for the normal distribution with mean $\mu = 0$ and variance $\sigma^2 = 1$.

to occur at a given point in the observation space. Recall, also, that the *cumulative distribution function (cdf)* $F(X)$ of a random variable X is defined for a number χ by:

$$F(\chi) = P(X \leq \chi) = \int_{-\infty}^{\chi} f(s)ds \tag{1}$$

where $f(s)$ is the likelihood associated with the random variable X obtained from the pdf f at s. The pdf for the normal distribution with mean $\mu = 0$ and variance $\sigma^2 = 1$ is shown in Figure 5.

The cdf of the normal distribution the same distribution with mean $\mu = 0$ and variance $\sigma^2 = 1$ is also shown in Figure 5. The cdf $P(X \leq \chi)$ is the probability that X is less than or equal to χ. It answers the question of "What is the likelihood of getting any load less than a specified load for a particular resource for a VM?" However, even though this metric may be useful, a better question to ask may be in the opposite order. "What is the maximum load y for a given likelihood z, such that $F(y) \leq z$?" This question can be answered by using the inverse cumulative distribution function.

The *inverse cumulative distribution function* (icdf) or *quantile function* returns the value below which random draws from the given cumulative distribution function would fall, $p * 100$ percent of the time. That is, it returns the value of χ such that

$$F(\chi) = Pr(X \leq \chi) = p \tag{2}$$

for a given probability $0 < p < 1$.

Using the icdf, we can specify a percentile value and obtain a corresponding load which can be passed to the assignment algorithm. Using the value from a high percentile will result in a high load being passed to the assignment algorithm, and tend to make the assignment more robust to random variation. Lower percentiles result in assignments more likely to become over loaded. Using the quantile function to decide which load to use means that the algorithm designer can incorporate any level of robustness or aggressiveness into the algorithm. Figure 5 shows the inverse cumulative distribution function for the normal distribution with mean $\mu = 0$ and variance $\sigma^2 = 1$. We call the load derived from the icdf value the *derived load*. The derived load is the load used by the bin packing algrorithm when a concrete value must be used.

IV. Initial Assignment Algorithms

Many companies and organizations do not use a structured approach to the initial placement of VMs onto servers. Even though a system administrator may view a summarization of the load for each VM, and place VMs onto servers using that summarization, we have not found any formalization of a model for VMAP, nor any algorithm meant to make an initial placement of VMs to servers. Because this is a new model, and the model derives its roots in the Bin Packing Problem, we present some well-known algorithms that solve the Bin Packing Problem for consideration. We will describe and compare four assignment algorithms in this paper–Worst Fit, First Fit, Permutation Pack [10] and Reordering Grouping Genetic Algorithm [21].

A. Worst Fit

The worst fit algorithm for bin packing considers each item in order. For each item, first, it considers only bins that already have at least one item placed in them. It places the item in the bin into which it fits which has the most amount of free space. If the item does not fit in any of the bins considered, it is placed into a new bin.

The worst fit algorithm is considered in this paper because of its tendency to leave a bit of space in every bin that it uses. This extra space may be helpful when observed loads on VMs exceed what the assignment algorithm expected.

Finding the emptiest bin is easy to do in the conventional Bin Packing Problem as each item only has one weight. The sum of the weights of all the items in any one bin can be used to give a number describing the fullness of a bin. However, finding the emptiest server in VMAP is not as obvious because each VM has multiple loads which it exerts on the server. In order to compare the emptiness of one server to another's, there must be a way to combine the different loads. We use a Euclidean distance metric to compare the amount of free space in two different bins.

B. First Fit

The first fit algorithm for bin packing is a way to place an initial set of VMs on servers. In the first fit packing algorithm, items are arranged in order (often decreasing order). Bins are also arranged in a list. Each item is then considered in order and placed into the first bin into which it fits.

C. Permutation Pack

Permutation Pack (PP) attempts to find items in which the largest w components are exactly ordered with respect to the ordering of the corresponding smallest elements in the current bin [10]. Let R_i denote the remaining space in the ith capacity of a particular bin. If $d = 2$ and $R_1 < R_2$, then we look for an item n such that $w_{n_1} < w_{n_2}$. If no item is found, the requirements are continually relaxed until one is found. One of the weaknesses of Permutation Pack is the running time. If

 ISBN: 978-1-61208-106-9

all permutations are considered, it runs in $O(d!\, n^2)$ where d is the number of resource types and n is the number of items to be packed. We refer the reader to Leinberger et al. [10] for further description of the algorithm.

D. Reordering Grouping Genetic Algorithm

Reordering Grouping Genetic Algorithm (RGGA) is a genetic algorithm that was developed for the Multi-Capacity Bin Packing Problem [21]. RGGA has been shown to solve the Multi-Capacity Bin Packing Problem quickly, developing good solutions. RGGA represents an instance of the bin packing problem not only as an assignment of items to bins, but also as a list of items to be first fit packed. RGGA's crossover operator is an exon shuffling crossover as defined by Rohlfshagen et al [13]. RGGA uses a mutation operator that swaps two items in the first fit list $\frac{1}{3}$ of the time, moves an item from one spot to another in the first fit list $\frac{1}{3}$ of the time, and eliminates a bin, reinserting the contents $\frac{1}{3}$ of the time. This last idea is the mutation operator used by Faulkenauer in his Grouping Genetic Algorithm [7].

RGGA was shown to find optimal solutions to the bin packing algorithm in fewer iterations than the leading genetic algorithms in literature. As well, RGGA was shown to generate very good solutions to even large problem sizes of the Multi-Capacity Bin Packing Problem.

V. Metrics to Determine Success

In this section, we investigate two different metrics to determine the level of success of an initial VM assignment. These two metrics are total number of servers used and the proportion of servers over capacity.

A. Number of Servers Used

The total *number of servers* used is defined as the number of servers upon which VMs are placed. This metric is useful in determining the tightness of a particular assignment. An assignment which places its VMs on fewer servers will likely save energy in the long run. Aggressive assignment algorithms often maximize this metric.

B. Proportion of Server Resources Over Capacity

A server resource is over capacity if VMs on the server request more of that resource than is available on the server. For example, if the sum of the total amount of RAM requested by all the VMs on a particular server is greater than the amount available on the server, then the RAM on the server would be considered over capacity. In order to calculate the proportion of server resources over capacity, we divide the sum of all total server resources over capacity by the sum of all total server resources. The algorithm for calculating the *proportion of servers over capacity* is shown in Equation 3.

$$\frac{\sum_{b_i \in B} \sum_{j \in b_i} F(\sum_{w_{j_k} \in \vec{w}_k} w_{j_k}, C_k)}{\sum_{b_i \in B} \sum_{j \in b_i} 1} \tag{3}$$

where $F(X, Y)$ returns 1 if X is greater than Y and 0 otherwise.

Conservative VM assignment algorithms perform worse with regard to this metric, because they, on average, have less allocated resources per server. If a particular VM uses more server resources than was allotted to it, the server might not be over capacity if the algorithm chose to leave extra room. Algorithms that are more conservative when assigning VMs also yield solutions with a greater total number of servers.

VI. Experimental Setup

Because the contribution of this paper is the model proposed for VMAP, we wish to validate this model in our results by showing that the modifications we made to the conventional Bin Packing Problem are indeed helpful in modeling VMAP.

In our experiments, we did exactly what we expect real system administrators to do with our work. We deployed various VMs to a cluster of computers, gathered data on resource utilization of these VMs, generated an assignment for these VMs to servers, and carried out that assignment with virtualization software. The VMs that we created were of the form such that 8-12 of them would fit on a physical server. Because deployments in real data centers normally have 8-12 VMs per physical server [6], we expect our results to generalize well to other clusters.

We use the data itself as a nonparametric statistical distribution. The mean of this distribution can be computed by finding the mean of the data. The variance of the distribution can be found by finding the variance of the data. This distribution can be sampled by picking a data point with uniform probability. The icdf of this distribution can be found for any percentile by sorting the data, multiplying the percentile used by the number of data points, and returning that particular data point.

With this nonparametric distribution for the load on each VM, we were able to generate new assignments of VMs to servers. We ran each assignment algorithm for varying icdf values. We show the predicted performance for varying icdf values in our results. After simulating the assignment of VMs to servers, we predicted the number of servers and proportion of servers over capacityfor the case if we actually carried out the assignment. In order to obtain our simulated metrics, we sampled from the load distribution for each VM and assigned loads to VMs from their distributions. Using this data, we obtained predicted results for what a assignment would be like if we carried out the assigning on the servers [1].

After obtaining predicted results for assigning VMs to servers, we then reassigned VMs to servers, averaged the results and analyzed how closely our model showed what happened in real life. We show the results of these new assignments in our results. We used the Kernel-Based Virtual Machine (KVM) kernel virtualization infrastructure, the qemu processor emulator, and the libvirt virtualization management tool. In our data set, we used VMs with varying CPU and RAM loads. We kept track of the loads on the VMs and recorded the observed CPU and RAM utilization on host servers.

Because it is an option that the system administrator can tweak, we also show the predicted performance for different

 ISBN: 978-1-61208-106-9

algorithms for varying icdf values. As mentioned in Section IV, raising the icdf value makes an assignment algorithm more conservative and lowering the icdf value makes an assignment algorithm more aggressive.

In our simulated experiments, we performed these steps:

1) The packing algorithm receives some type of distribution over the load for each virtual machine it needs to pack.
2) The packing algorithm derives a specific load using the distribution from step 1 and the icdf value used.
3) The packing algorithm develops an assignment of virtual machines to servers.
4) One specific load for each virtual machine is sampled from the load distribution for that virtual machine. This load is assigned to that virtual machine.
5) Metrics are gathered.
6) Steps 1-4 are repeated as many times as needed to achieve statistical significance for the test.

When we implemented RGGA, we used a maximum function evaluation count of 7500, a crossover rate of 0.8, and a mutation rate of 0.1. In the event that two individuals do not crossover, one of them is picked randomly and added directly to the next population. If an individual is not mutated, it is simply added as is to the next generation.

VII. Results

In our results, we wish to validate the probabilistic model proposed in Section III and the algorithms we proposed in Section IV. We will divide presentation of results in three parts. First, we will show how system administrators can use assignment algorithms with varying icdf values. We then analyze and present graphs that show in a practical standpoint number of servers used and the proportion of servers over capacity. Lastly, we analyze how closely our predictive model resembles what happens on real hardware by repacking VMs to servers.

Even though we do not directly incorporate probabilistic loads into any assignment algorithm proposed in this paper, we show results that help system administrators to indirectly incorporate probabilistic results into the assignment algorithm picked by applying the ideas presented in Section III-A. We systematically increased the icdf value and subsequently the derived load from the icdf value. Increasing this value increases the derived loads on VMs which the algorithm uses. As discussed earlier, lower values of percentile yield more conservative assignment algorithms and higher value of percentile yield more aggressive assignment algorithms.

Figure 6 shows the proportion of servers over capacity while Figure 7 shows the number of servers found by the different algorithms. The independent variable in both graphs is the icdf value used. Even though we present both figures separately, they must be interpretted together. One shows the number of servers used, while the other shows the proportion of servers over capacity. Algorithms which tend to use fewer bins will tend to look better in Figure 7, but look worse in Figure 6.

The icdf value is merely a parameter used to determine both the number of servers used and the proportion of servers

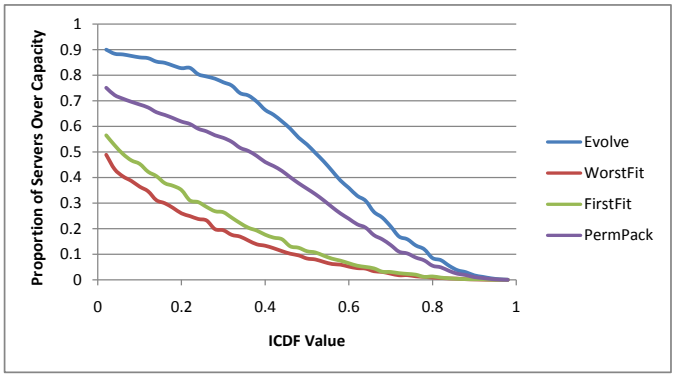


Fig. 6. A comparison of the proportion of servers over capacity.

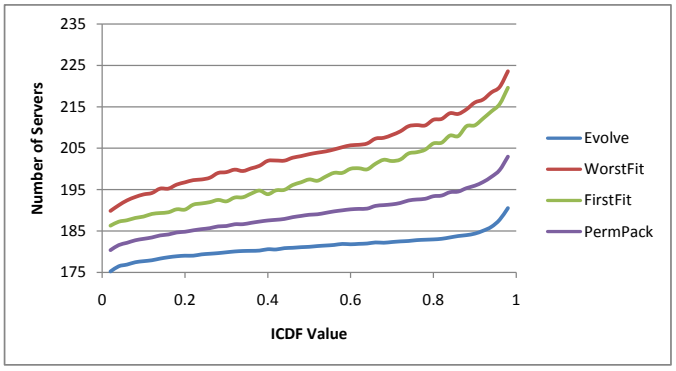


Fig. 7. A comparison of the the number of servers found.

over capacity. The icdf value used is really irrelevant because the icdf value picked by system administrators is a function of the number of servers used and the proportion of servers over capacity. Instead of showing this graph, we wish to combine Figures 6 and 7 so that we can see this type of interaction between the proportion of servers over capacity and the number of servers used.

In order to simplify our analysis, we joined the proportion of servers over capacity with the number of servers found using the percentile values to produce a joint graph. Using this graph, Figure 8, a system administrator can choose how many

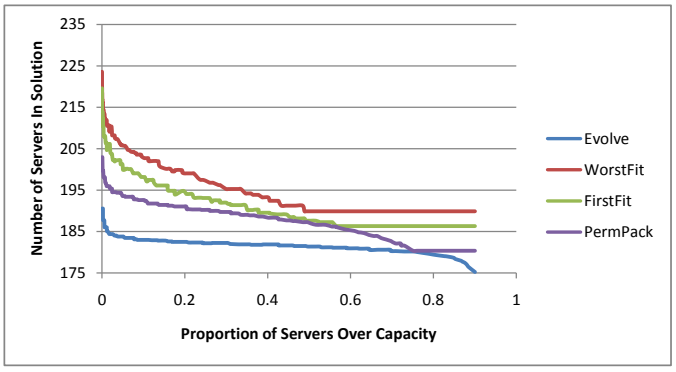


Fig. 8. A comparison of the proportion of servers over capacity with the number of servers found.

 ISBN: 978-1-61208-106-9

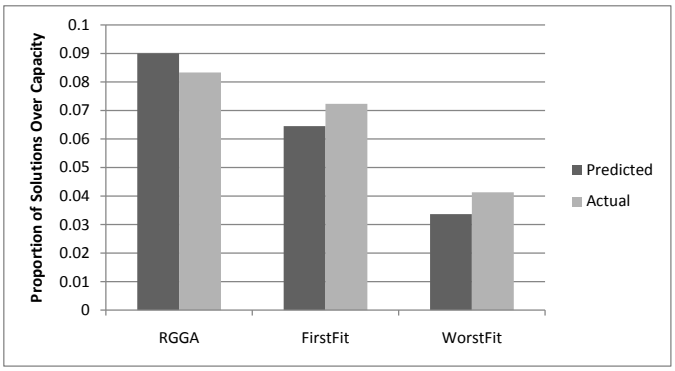


Fig. 9. A comparison of the proportion of solutions over capacity that our algorithm predicted and also the measured proportion of solutions over capacity when deploying real VMs to servers for the 80th cdf percentile.

servers on average will be in utilization or what proportion over capacity they are willing to tolerate in order to get the other parameter. This graph shows that for this particular configuration, RGGA

A. Repacking to Servers

Lastly, in order to validate the model we generated, we used the assignments generated to make assignments of real VMs to servers. We measured the proportion of server resources over capacity for all three assignment algorithms at the 80th cdf percentile. We show our results in Figure 9. The predicted proportion of server resources over capacity all algorithms is very close to the actual proportion of server resources over capacity measured when deploying VMs to servers. This gives validity to the model we suggest in this paper.

VIII. Conclusion

A novel model for Virtual Machine Assignment Problem is proposed. This model uses ideas from the conventional Bin Packing Problem, where servers are bins and VMs are items, with two variations. First, it allows multiple weights for each item and multiple capacities for each bin. The sum of all the weight of any one type in any bin must be less than that corresponding capacity in that bin. Second, our model proposes that VMs have probabilistic loads. The probabilistic loads should be incorporated into the assignment algorithm for best results. We, show the feasibility of using probabilistic loads with the assignment algorithm by modifying three known packing algorithms.

Adding probability theory helps system administrators to pick the correct percentile value representing the spot on the inverse cumulative distribution function representing the load of a resource. Small values for the icdf value yield more conservative assignment algorithms while larger values for the icdf value yield more aggressive assignment algorithms.

For the problems which we investigated, it seems that optimization algorithms like RGGA perform well.

References

[1] Virtualization datasets. http://aml.cs.byu.edu/ davidw/datasets, July 2010.

[2] F. Glover A. C. F. Alvim, C. C. Ribeiro and D. J. Aloise. A hybrid improvment heuristic for the one-dimensional bin packing problem. *Journal of Heuristics*, 10:4–27, 2004.

[3] Sandip Agarwala, Fernando Alegre, Karsten Schwan, and Jegannathan Mehalingham. E2eprof: Automated end-to-end performance management for enterprise systems. In *DSN '07: Proceedings of the 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, pages 749–758, Washington, DC, USA, 2007. IEEE Computer Society.

[4] Michael Bailey, Timothy Rostrom, and J. Ekstrom. Operating system power dependencies. In *Int. CMG Conference*, pages 15–20, 2008.

[5] Gong Chen, Wenbo He, Jie Liu, Suman Nath, Leonidas Rigas, Lin Xiao, and Feng Zhao. Energy-aware server provisioning and load dispatching for connection-intensive internet services. In *NSDI'08: Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation*, pages 337–350, Berkeley, CA, USA, 2008. USENIX Association.

[6] IT Knowledge Exchange. Virtual machines per server: A viable metric for hardware selection? http://itknowledgeexchange.techtarget.com/server-farm/virtual-machines-per-server-a-viable-metric-for-hardware-selection/, August 2008.

[7] E. Faulkenaeur. A hybrid grouping genetic algorithm for bin packing. *Journal of Heuristics*, 2:5–30, 1996.

[8] Narendra Karmarkar and Richard M. Karp. An efficient approximation scheme for the one-dimensional bin-packing problem. *Foundations of Computer Science, 1982. SFCS '08. 23rd Annual Symposium on*, pages 312–320, November 1982.

[9] J. G. Koomey. Estimating total power consumption by servers in the u.s. and the world. *Presented at the EPA stakeholder workshop at Santa Clara Convention Center*, February 2007.

[10] William Leinberger, George Karypis, and Vipin Kumar. Multi-capacity bin packing algorithms with applications to job scheduling under multiple constraints. In *ICPP '99: Proceedings of the 1999 International Conference on Parallel Processing*, page 404, Washington, DC, USA, 1999. IEEE Computer Society.

[11] M. Mitchell. *An introduction to genetic algorithms*. MIT Press, 1998.

[12] Ripal Nathuji and Karsten Schwan. Virtualpower: coordinated power management in virtualized enterprise systems. In *SOSP '07: Proceedings of twenty-first ACM SIGOPS symposium on Operating systems principles*, pages 265–278, New York, NY, USA, 2007. ACM.

[13] P. Rohlfshagen and J. Bullinaria. A genetic algorithm with exon shuffling crossover for hard bin packing problems. *Proceedings of Genetic And Evolutionary Computation Conference*, 9:1365–1371, 2007.

[14] Louis P. Slothouber and Ph. D. A model of web server performance. In *In Proceedings of the Fifth International World Wide Web Conference*, 1996.

[15] Ying Song, Hui Wang, Yaqiong Li, Binquan Feng, and Yuzhong Sun. Multi-tiered on-demand resource scheduling for vm-based data center. In *CCGRID '09: Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid*, pages 148–155, Washington, DC, USA, 2009. IEEE Computer Society.

[16] Ying Song, Yanwei Zhang, and Yuzhong Sun. Utility analysis for internet-oriented server consolidation in vm-based data centers. *Cluster Computing, IEEE International Conference on*, 0:xvii, 2009.

[17] Shekhar Srikantaiah, Aman Kansal, and Feng Zhao. Energy aware consolidation for cloud computing. In *Proceedings of HotPower '08 Workshop on Power Aware Computing and Systems*. USENIX, December 2008.

[18] Mark Stillwell, David Schanzenbach, Frederic Vivien, and Henri Casanova. Resource allocation using virtual clusters. In *CCGRID '09: Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid*, pages 260–267, Washington, DC, USA, 2009. IEEE Computer Society.

[19] C. W. Szeto. On the modeling of www request arrivals. In *ICPP '99: Proceedings of the 1999 International Workshops on Parallel Processing*, page 248, Washington, DC, USA, 1999. IEEE Computer Society.

[20] VMware. Vmware green it energy efficiency, reduce energy costs with virtualization. http://nvd.nist.gov/nvd.cfm?cvename=CVE-2008-1368, September 2009.

[21] David Wilcox, Andrew McNabb, Kevin Seppi, and Kelly Flanagan. Virtual machine assignment with a reordering grouping genetic algorithm. *To be submitted to Congress on Evolutionary Computation*, 2011.

 ISBN: 978-1-61208-106-9

Handling Confidential Data on the Untrusted Cloud: An Agent-based Approach

Ernesto Damiani
Department of Information Technology
Università degli Studi di Milano
Milano, Italy
ernesto.damiani@unimi.it

Francesco Pagano
Department of Information Technology
Università degli Studi di Milano
Milano, Italy
francesco.pagano@unimi.it

***Abstract*— Cloud computing allows shared computer and storage facilities to be used by a multitude of clients. While cloud management is centralized, the information resides in the cloud and information sharing can be implemented via off-the-shelf techniques for multiuser databases. Users, however, are very diffident for not having full control over their sensitive data. Untrusted database-as-a-server techniques are neither readily extendable to the cloud environment nor easily understandable by non-technical users. To solve this problem, we present an approach where agents share reserved data in a secure manner by the use of simple grant and revoke permissions on shared data.**

Keywords - Information sharing; privacy; distributed data; cloud computing; multi-agent systems.

I. INTRODUCTION

Cloud computing is the commercial evolution of grid computing [21]; it provides users with readily available, pay-as-you-go computing and storage power, allowing them to dynamically adapt their IT (Information Technology) costs to their needs. In this fashion, users need neither costly competence in IT system management or huge investments in the start-up phase in preparation for future growth.

While the cloud computing concept is drawing much interest, several obstacles remain to its widespread adoption including:

- Current limits of ICT infrastructure: availability, reliability and quality of service;
- Different paradigm of development of web applications with respect to those used for desktop applications;
- Privacy risks for confidential information residing in the cloud.

Hopefully, the first obstacle will diminish over time, thanks to the increasingly widespread availability of the network; the second will progressively disappear by training new developers and retraining the older; the third issue however, is far from being solved and may impair very seriously the real prospects of cloud computing.

In this paper, we illustrate some techniques for providing data protection and confidentiality in outsourced databases (Section II) and then we analyze some possible pitfalls of these techniques in Cloud Computing (Section III), which bring us to propose a new solution based on multi-agent systems (Section IV).

II. THE PROBLEM OF PRIVACY

The cloud infrastructure can be accessible to public users (Public Cloud) or only to those operating within an organization (Private Cloud) [1]. Generally speaking, external access to shared data held by the cloud goes through the usual authentication authorization and communication phases. The access control problem is well-known in the database literature and available solutions guarantee a high degree of confidence.

However, the requirement that outsourced data cannot be accessed or altered by the maintainer of the datastore is not met as easily, especially on public clouds like Google App Engine for Business, Microsoft Azure Platform or Amazon EC2 platform.

Indeed, existing techniques for managing the outsourcing of data on untrusted database servers [11] [12] cannot be straightforwardly applied to public clouds, due to several reasons:

- The physical structure of the cloud is, by definition, undetectable from the outside: who is really storing the data?
- The user often has no control over data replication, i.e., how many copies exist (including backups) and how are they managed?
- The lack of information on the geographical location of data (or its variation over time) may lead to jurisdiction conflicts when different national laws apply.

In the next section, we will briefly summarize the available techniques for data protection on untrusted servers, and show how they are affected by the problems outlined above.

A. Data Protection

To ensure data protection in outsourcing, the literature reports three main techniques [4]:

- Data encryption [13];
- Data fragmentation and encryption [14];
  - non-communicating servers [15][16];
  - unlinkable fragments [17];
- Data fragmentation with owner involvement [18].

1) Data encryption

To prevent unauthorized access by the datastore manager (DM) managing the outsourced RDBMS (Relational Data Base Management Systems), the data is stored encrypted.

 ISBN: 978-1-61208-106-9

Obviously, the encryption keys are not known to the DM and they are stored apart from the data. The RDBMS receives an encrypted database and it works on meaningless bit-streams that only the clients, who hold the decryption keys, can interpret correctly.

Note that decryption keys are generated and distributed to trusted clients by the data owner or by a trusted delegate.

Encryption can occur with different levels of granularity: field, record, table, db. For efficiency reasons, normally, the level adopted is the record (tuple in relational databases).

Of course, because the data is encrypted, the DBMS cannot index it based on plaintext and therefore it cannot resolve all queries. Available proposals tackle this problem by providing, for each (encrypted) field to be indexed, an additional indexable field, obtained by applying a non-injective transformation *f* to plaintext values (e.g., a hashing of the field's content). This way, queries can be performed easily and with equality constraints, although with a precision < 1 (to prevent statistical data mining). The trusted client, after receiving the encrypted result set for the query, will decrypt and exclude spurious tuples. In this setting, however, it is difficult to answer range queries, since *f* in general will not preserve the order relations of the original plaintext data. Specifically, it will be impossible for the outsourced RDBMS to answer range queries that cannot be reduced to multiple equality conditions (e.g., *1<=x<=3* can be translated into *x=1 or x=2 or x=3*). In literature, there are several proposals for *f*, including:

1. *Domain partitioning* [22]*:* the domain is partitioned into equivalence classes, each corresponding to a single value in the codomain of *f*;
2. *Secure hashing* [11]: secure one-way hash function, which takes as input the clear values of an attribute and returns the corresponding index values. *f* must be deterministic and non-injective.

To handle range queries, a solution, among others, is to use an encrypted version of a B ± tree to store plaintext values, and maintain the values order. Because the values have to be encrypted, the tree is managed at the Client side and it is read-only in the Server side.

2) Data fragmentation

Normally, of all the outsourced data, only some columns and/or some relations are confidential, so it is possible to split the outsourced information in two parts, one for confidential and one for public data. Its aim is to minimize the computational load of encryption/decryption.

a) Non-communicating servers

In this technique, two *split databases* are stored, each in a different untrusted server (called, say, S_1 and S_2). The two untrusted servers have to be independent and non-communicating, so they cannot ally themselves to reconstruct the complete information. In such situation, the information may be stored in plaintext in each server.

With this approach, each Client query need be decomposed in two subqueries: one for S_1 and one for S_2. The resulting sets have to be related and filtered, later, at Client level.

b) Unlinkable fragments

In reality, it is not easy to ensure that split servers do not communicate; therefore the previous technique may be inapplicable. A possible remedy is to divide information in two or more fragments. Each fragment contains all the fields of original information, but some are in clear while the others are encrypted. To protect encrypted values from frequency attacks, a suitable *salt* is applied to each encryption. Fragments are guaranteed to be unlinkable (i.e., it is impossible to reconstruct the original relation and to determine the sensitive values and associations without the decrypting key). These fragments may be stored in one or more servers.

Each query is then decomposed in two subqueries:

- The first, on the Server, chooses a fragment (all fragments contain the entire information) and selects tuples from it according to clear values and returns a result set where some fields are encrypted;
- The second, on Client (only if encrypted fields are involved in the query), decrypts the information and removes the spurious tuples according to encrypted values.

3) Data fragmentation with owner involvement

Another adaptation of non-communicating servers consists of storing locally the sensitive data and relations, while outsourcing the storage of the generic data. So, each tuple is split in a server part and in a local part, with the primary key in common. The query is then resolved as shown above.

B. Selective access

In many scenarios, access to data is selective, with different users enjoying different views over the data. Access can discriminate between read and write of a single record or only a part of it.

An intuitive way to handle this problem is to encrypt different portions of data with different keys that are then distributed to users according to their access privileges. To minimize overhead we want that:

- No more than one key is released to each user;
- Each resource is encrypted not more than once.

To achieve these objectives, we can use a hierarchical organization of keys. Basically, users with the same access privileges are grouped and each resource is encrypted with the key associated with the set of users that can access it. In this way, a single key can be possibly used to encrypt more than one resource.

1) Dynamic rights management

Should the user's rights change over time (e.g., the user changes department) it is necessary to remove that user from a group/role as follows:

- Encrypt data by a new key;
- Remove the original encrypted data;
- Send the new key to the rest of the group.

Note that these operations must be performed by data owner because the untrusted DBMS has no access to the keys. This active role of the data owner goes somewhat

 ISBN: 978-1-61208-106-9

against the reasons for choosing to outsource data in the first place.

a) Temporal key management

An important issue, common to many access control policies, concerns time-dependent constraints of access permissions. In many real situations, it is likely that a user may be assigned to a certain role or class for only a certain period of time. In such case, users need a different key for each time period. A time-bound hierarchical key assignment scheme is a method to assign time-dependent encryption keys and private information to each class in the hierarchy in such a way that key derivation also depends on temporal constraints. Once a time period expires, users in a class should not be able to access any subsequent keys if not authorized to do so [7].

b) Database replica

In [5], the authors, exploiting the never ending lower price-per-byte, propose to replicate *n* times the source database, where *n* is the number of different roles having access to the database. Each database replica is a view, entirely encrypted using the key created for the corresponding role. Each time that a role is created, the corresponding view is generated and encrypted with a new key expressly generated for the newly created role. Users do not own the real key, but receive a token that allows them to address a cipher demand to a set KS of key servers on the cloud.

C. A document base sample: Crypstore

An example of data protection implementation by data encryption is Crypstore. It is a non-transactional architecture for the distribution of confidential data. The Storage Server contains data in encrypted form, so it cannot read them. User who wants to access data is authenticated at the Key Servers with the certificate issued by the Data Administrator and requires the decryption key. The Key Servers are *N* and, to ensure that none of them knows the whole decryption key, each of them contains only a part of the encryption key. To rebuild the key, only *M* (<*N*) parts of key are needed; redundancy provides greater robustness to failures and attacks (e.g., Denial of Service attacks).

In practice, it is an application of the time-honored "divide and conquer" technique, where data is separated from decryption keys.

Here the privacy is not entirely guaranteed because, theoretically at least, the owner of Key Servers and the Storage Server may agree to overcome the limitations of the system. The only way to exclude the (remote) possibility is to have trusted Key Servers, but if so, it would be useless to distinguish the two structures and we could take data directly, as plaintext, to a trusted storage. Such criticism applies however only in theory because, in practice, the probability of such an agreement decreases with the number of players involved.

III. PRIVACY WITHIN THE CLOUD

All techniques discussed above are based on data encryption and/or data fragmentation using full separation of roles and of execution environments between the user and the datastore (and possibly the keystore) used to manage the outsourced data.

Let us now compare the assumptions behind such techniques with two of the basic tenets of current cloud computing architectures: data and applications being on the "same side of the wall", and data being managed via semantic datastores rather than by a conventional RDBMS.

A. On the same side of the wall

Ubiquitous access is a major feature of cloud computing architectures. It guarantees that cloud application users will be unrestrained by their physical location (with internet access) and unrestrained by the physical device they use to access the cloud.

To satisfy the above requirements (in particular the second), we normally use thin clients, which run cloud applications remotely via a web user interface.

The three main suppliers of Public Cloud Infrastructure (Google App Engine for Business, Amazon Elastic Compute Cloud and Windows Azure Platform) all include a datastore, and an environment for remote execution summarized in Tables I and II:

TABLE I. DATASTORE SOLUTIONS USED BY PUBLIC CLOUDS

Environment	Datastore
Google	Bigtable
Amazon	IBM DB2 IBM Informix Dynamic Server Microsoft SQLServer Standard 2005 MySQL Enterprise Oracle Database 11g Others installed by users
Microsoft	Microsoft SQL Azure

TABLE II. EXECUTION ENVIRONMENTS USED BY PUBLIC CLOUDS

Environment	Execution environment
Google	J2EE (Tomcat + GWT) Python
Amazon	J2EE (IBM WAS, Oracle WebLogic Server) and others installed by users
Microsoft	.Net

In all practical scenarios, public cloud suppliers handle both data and application management.

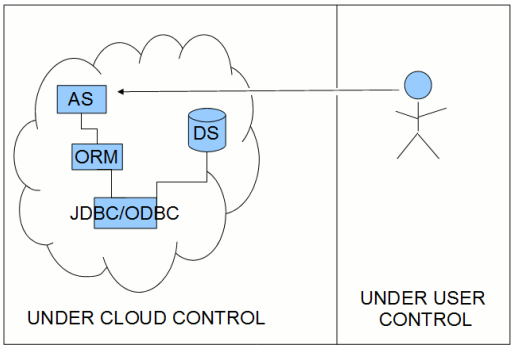


Figure 1. The wall

 ISBN: 978-1-61208-106-9

If the cloud supplier is untrustworthy, she can intercept communications, modify executable software components (e.g., using aspect programming), monitor the user application memory, etc.

Hence, available techniques for safely outsourcing data to untrusted DBMS no longer guarantees the confidentiality of data outsourced to the cloud.

The essential point consists in having the data and the user interface application logic *on the same side of the wall*. This is a major difference w.r.t. the outsourced database scenarios, where presentation was handled by trusted clients. In the end, the data must be presented to the user in an intelligible and clear form; that is the moment when a malicious agent operating in the cloud has more opportunities to intercept the data. To prevent unwanted access to the data at presentation time, it would be appropriate moving the presentation logics off the cloud to a trusted environment that may be an intranet or, at the bottom level, a personal computer.

However, separating data (which would stay in the cloud) from the presentation logics may enable the creation of local copies of data, and lead to an inefficient cooperation between the two parts.

B. *Semantic datastore*

Cloud computing solutions largely rely on semantic (non-relational) DBMS. These systems do not store data in tabular format, but following the natural structure of objects. After more than twenty years of experimentation (see, for instance, [8] for the Galileo system developed at the University of Pisa), today, the lower performance of these systems is no longer a problem. In the field of cloud computing, there is a particular attention to Google Bigtable.

"*Bigtable is a distributed storage system for managing structured data that is designed to scale to a very large size: petabytes of data across thousands of commodity servers. In many ways, Bigtable resembles a database: it shares many implementation strategies with databases.*" [9]

With a semantic datastore like Bigtable, there is a more strict integration between in-memory data and stored-data; they are almost indistinguishable from programmer viewpoint. There are not distinct phases when the program loads data from disk into main memory or, in the opposite direction, when program serialize data on disk. Applications do not even know where data is stored, as it is scattered over the cloud.

In such a situation, the data outsourcing techniques discussed before cannot be applied directly, because they were designed for untrusted RDBMS.

IV. OUR APPROACH

We are now ready to discuss our new approach to the problem of cloud data privacy. We build over the notion introduced in [5] of defining a view for every user group/role, but we prevent performance degradation by keeping all data views in the user environment.

Specifically, we atomize the couple application/database, providing a copy per user. Every instance runs locally, and maintains only authorized data that is replicated and synchronized among all authorized users.

In the following subsections we will analyze our solution in detail.

A. *Information sharing by multi-agent system*

We will consider a system composed of:

1. Local agents distributed at client side;
2. A central synchronization point.

1) *The model*

In the following, we will use the term *dossier* to indicate a set of correlated information. Our data model may be informally represented by the diagram in Figure 4.

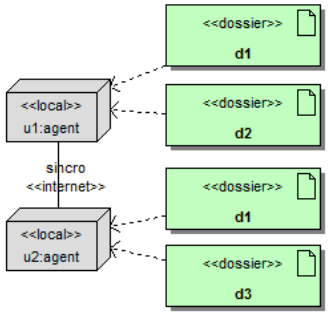


Figure 2. The model

In the model, each node represents a local, single-user application/database dedicated to an individual user (u_n). The node stores only the dossiers that u_n owns. Shared dossiers (in this example, d_1) are replicated on each node. When a node modifies a shared dossier, it must synchronize, also using heuristics and learning algorithms, with the other nodes that hold a copy of it. Below we give a simple SWOT analysis of this idea.

2) *Strength/Opportunities*

- Unrestrained individual nodes, that can also work offline (with deferred synchronization);
- Simplicity of data management (single user);
- Completeness of local information.

To understand the last point, suppose that the user u_n wants to know the number of the dossier she is treating. In a classic intranet solution, where dossiers would reside on their owners' servers, in addition to its database, u_n should examine the data stores of all other collaborating users. With our solution, instead, u_n can simply perform a local query because the dossiers are replicated at each client.

3) *Weaknesses/Threats*

- Complexity of deferred synchronization schemes [19];
- Necessity to implement a mechanism for grant/revoke and access control permissions.

This last point is particularly important and it deserves further discussion:

- As each user (except the data owner) may have partial access to a dossier, each node contains only the allowed portion of the information;

 ISBN: 978-1-61208-106-9

- Authorization, i.e., granting to a user u_j access to a dossier d_k, can be achieved by the data owner simply by transmitting to the corresponding node only the data it is allowed to access;
- The inverse operation will be made in the case of a (partial or complete) revocation of access rights. An obvious difficulty lies in ensuring that data, once revoked, is no longer available to the revoked node. This is indeed a moot point, as it is impossible – whatever the approach - to prevent trusted users from creating local copies of data while they are authorized and use them after revocation.

B. *Proposed solution*

We are now ready to analyze in detail our solution. To simplify the discussion, we introduce the following assumptions:

- Each dossier has only one owner;
- Only the dossier's owner can change it.

Those assumptions allow the use of an elementary cascade synchronization in which the owner will submit the changes to the receivers.

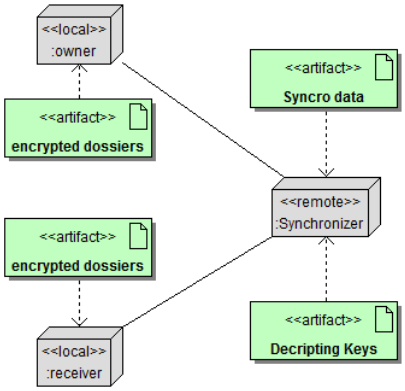


Figure 3. Deployment diagram of multi-agent system

Our solution consists of two parts: a trusted client agent and a remote untrusted synchronizer.

The client maintains local data storage where:

- The dossiers whom he owns are (or at least can be) stored as plaintext;
- The others, instead, are encrypted, each with a different key.

The Synchronizer stores the keys to decrypt the shared dossiers owned by the local client and the modified dossiers to synchronize.

When another client needs to decrypt a dossier, he must connect to the Synchronizer and obtain the corresponding decryption key.

The data and the keys are stored in two separate entities and therefore none can access information without the collaboration of the other part.

1) *Structure*

From the architectural point of view, we divide our components into two packages, a local (client agent), which contains the dossier and additional information such as access lists, and a remote (global synchronizer), which contains the list of dossiers to synchronize, their decryption keys and the public keys of clients.

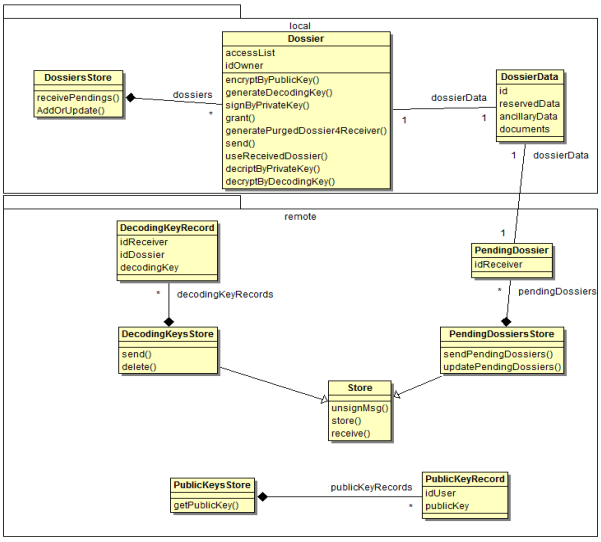


Figure 4. Class view

2) *Grant*

An owner willing to grant rights on a dossier must follow the following sequence:

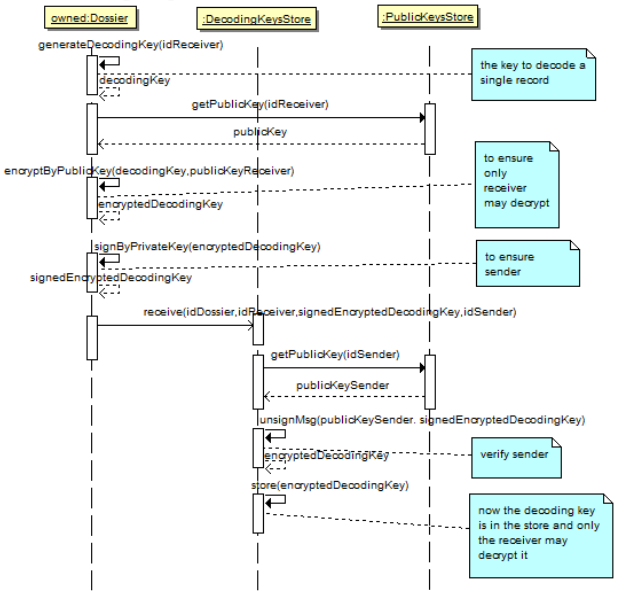


Figure 5. Grant sequence

Namely, for each receiver, the owner:

- generates the decryption key
- encrypts it with the public key of the receiver to ensure that others cannot read it
- signs it with its private key to ensure its origin
- sends it to the Synchronizer, which verifies the origin and adds it to the storage of the decoding

 ISBN: 978-1-61208-106-9

keys. The key is still encrypted with the public key of the receiver, so only the receiver can read it.

3) Send

When an owner modifies a dossier, she sends it to the Synchronizer following this sequence:

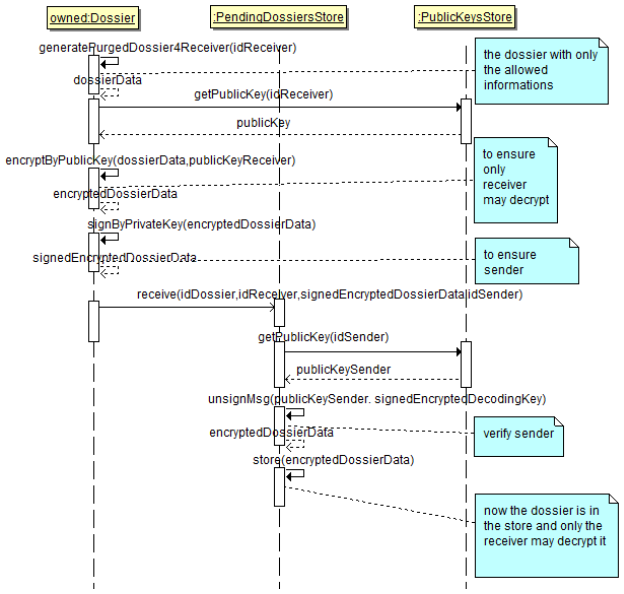


Figure 6. Send sequence

For each receiver, the owner:

- generates a "pending dossier" by removing information that the receiver should not have access to;
- encrypts it with the public key of the receiver to ensure that others cannot read it;
- signs with his own private key to certificate its origin;
- sends it to the Synchronizer, which verifies the origin and adds it to the storage of "pending dossiers". Again, the dossier is still encrypted with the public key of the receiver, so only the receiver can read it.

4) Receive

Periodically, each client updates un-owned dossiers by following this sequence:

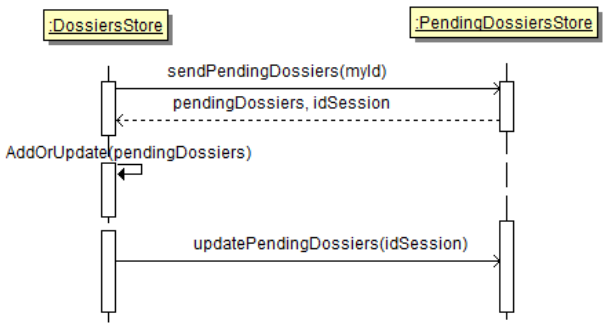


Figure 7. Receive sequence

Each client:

- requests the Synchronizer the "pending dossiers";
- modifies the local storage;
- removes from the Synchronizer the received dossiers.

5) Use

When a client needs to use an unowned (encrypted) dossier, the following sequence is used:

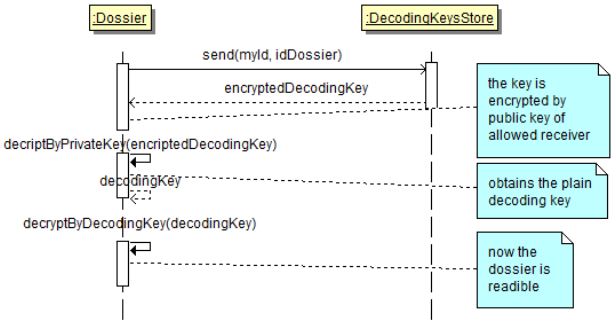


Figure 8. Use sequence

The client:

- asks the Synchronizer for the decryption key (that is encrypted by his public key);
- decrypts it with its private key;
- decrypts the dossier by the resulting decryption key.

If the decryption key does not exist, two options are available:

- the record is deleted from the local datastore because a revoke happened;
- the record remains cached (encrypted) into the local datastore because the access rights could be restored.

6) Revoke

To revoke access to a receiver, it is sufficient to delete the corresponding decryption key from the Synchronizer:

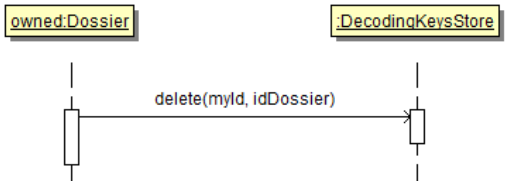


Figure 9. Revoke sequence

7) Implementation

We are currently implementing the proposed solution using an IMDB (in-memory database), such HyperSql (www.hsqldb.org). An in-memory database (IMDB also known as main memory database system or MMDB) is a database management system that primarily relies on main memory for computer data storage.

A HyperSql db consists of a text file containing sql instructions to:

- create structure (tables, indexes, etc.);
- populate tables.

At DBMS startup, this file is read and HyperSql creates a data model of the db into memory. At closing, the data

 ISBN: 978-1-61208-106-9

model is serialized on the disk (actually also intermediate writes in a log file occur, to minimize the risk of data loss for sudden failure). The implementation of our solution, therefore, will consist in rewriting the load and save operations. The load function need implement the above-mentioned sequence.

8) Future work

In the next future, we must deepen the synchronization algorithm [23], benchmark the performance in a system under stress and use a cache of decoding time-bounded keys [6] to allow users to work offline.

V. CONCLUSIONS AND OUTLOOK

In this paper, we discussed the applicability of outsourced DBMS solutions to the cloud and provided the outline of a simple yet complete solution for managing confidential data in public clouds.

We are fully aware that a number of problems remain to be solved. A major weakness of any data outsourcing scheme is the creation of local copies of data after it has been decrypted. If a malicious client decrypts data and then it stores the resulting plaintext data in a private location, the protection is broken, as the client will be available to access its local copy after being revoked. In [20], obfuscated web presentation logic is introduced to prevent client from harvesting data. This technique, however, exposes plaintext data to cloud provider. The manager of plaintext data is always the weak link in the chain and any solution must choose whether to trust the client-side or the server-side.

Another issue concerns the degree of trustworthiness of the participants. Indeed, untrusted Synchronizer never holds plaintext data; therefore it does not introduce an additional Trusted Third Party (TTP) with respect to the solutions described at the beginning of the paper. However, we need to trust the Synchronizer to execute correctly the protocols explained in the paper. This is a determining factor that our technique shares with competing solutions and, although an interesting topic, it lies beyond the scope of this paper.

ACKNOWLEDGMENT

We would like to thanks Sabrina De Capitani di Vimercati and Pierangela Samarati for providing us with their seminal paper [4].

This work was partly founded by the European Commission under the project SecureSCM (contract n. FP7-213531).

REFERENCES

[1] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy H. Katz, Andy Konwinski, Gunho Lee, David A. Patterson, Ariel Rabkin, Ion Stoica, and Matei Zaharia: “A view of cloud computing”, Commun. ACM 53(4), pp. 50-58 (2010)

[2] C. Jackson, D. Boneh, and J.C. Mitchell: “Protecting Browser State from Web Privacy Attacks”, 15th International World Wide Web Conference (WWW 2006), Edinburgh, May, 2006.

[3] Philip A. Bernstein, Fausto Giunchiglia, Anastasios Kementsietsidis, John Mylopoulos, Luciano Serafini, and Ilya Zaihrayeu: “Data Management for Peer-to-Peer Computing : A Vision”, WebDB 2002, pp. 89-94

[4] Pierangela Samarati and Sabrina De Capitani di Vimercati: “Data protection in outsourcing scenarios: issues and directions”, ASIACCS 2010, pp. 1-14

[5] Nadia Bennani, Ernesto Damiani, and Stelvio Cimato: “Toward cloud-based key management for outsourced databases”, SAPSE 2010, draft

[6] Mikhail J. Atallah, Marina Blanton, and Keith B. Frikken: “Incorporating Temporal Capabilities in Existing Key Management Schemes”, ESORICS 2007, pp. 515-530

[7] Alfredo De Santis, Anna Lisa Ferrara, and Barbara Masucci: “New constructions for provably-secure time-bound hierarchical key assignment schemes”, Theor. Comput. Sci. 407, pp.213-230 (2008)

[8] Antonio Albano, Giorgio Ghelli, M. Eugenia Occhiuto, and Renzo Orsini: “Object-Oriented Galileo”, On Object-Oriented Database System 1991, pp. 87-104

[9] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach, Michael Burrows, Tushar Chandra, Andrew Fikes, and Robert Gruber: “Bigtable: A Distributed Storage System for Structured Data”, OSDI 2006, pp. 205-218

[10] Victor R. Lesser: “Encyclopedia of Computer Science”, 4th edition. John Wiley and Sons Ltd. 2003, pp.1194–1196

[11] Ernesto Damiani, Sabrina De Capitani di Vimercati, Sushil Jajodia, Stefano Paraboschi, and Pierangela Samarati:“Balancing confidentiality and efficiency in untrusted relational DBMSs”,ACM Conference on Computer and Comm. Security 2003, pp.93-102

[12] Ernesto Damiani, Sabrina De Capitani di Vimercati, Mario Finetti, Stefano Paraboschi, Pierangela Samarati, and Sushil Jajodia: “Implementation of a Storage Mechanism for Untrusted DBMSs”, IEEE Security in Storage Workshop 2003, pp. 38-46

[13] Sabrina De Capitani di Vimercati, Sara Foresti, Stefano Paraboschi, and Pierangela Samarati: “Privacy of outsourced data”, In Alessandro Acquisti, Stefanos Gritzalis, Costos Lambrinoudakis, and Sabrina De Capitani di Vimercati: Digital Privacy: Theory, Technologies and Practices. Auerbach Publications (Taylor and Francis Group) 2007

[14] Valentina Ciriani, Sabrina De Capitani di Vimercati, Sara Foresti, Sushil Jajodia, Stefano Paraboschi, and Pierangela Samarati: “Fragmentation and Encryption to Enforce Privacy in Data Storage”, ESORICS 2007, pp. 171-186

[15] Richard Brinkman, Jeroen Doumen, and Willem Jonker: “Using Secret Sharing for Searching”, in Encrypted Data. Secure Data Management 2004, pp. 18-27

[16] Ping Lin and K. Selçuk Candan: “Secure and Privacy Preserving Outsourcing of Tree Structured Data”, Secure Data Management 2004, pp. 1-17

[17] Valentina Ciriani, Sabrina De Capitani di Vimercati, Sara Foresti, Sushil Jajodia, Stefano Paraboschi, and Pierangela Samarati: “Combining fragmentation and encryption to protect privacy in data storage”, ACM Trans. Inf. Syst. Secur. 13(3): (2010)

[18] Valentina Ciriani, Sabrina De Capitani di Vimercati, Sara Foresti, Sushil Jajodia, Stefano Paraboschi, and Pierangela Samarati: “Keep a Few: Outsourcing Data While Maintaining Confidentiality”, ESORICS 2009, pp. 440-455

[19] Miseon Choi, Wonik Park, and Young-Kuk Kim: “A split synchronizing mobile transaction model”, ICUIMC 2008, pp.196-201

[20] Henk C. A. van Tilborg: “Encyclopedia of Cryptography and Security”, Springer 2005

[21] Ian T. Foster, Yong Zhao, Ioan Raicu, and Shiyong Lu: “Cloud Computing and Grid Computing 360-Degree Compared CoRR”, abs/0901.0131: (2009)

[22] Hakan Hacigümüs, Balakrishna R. Iyer, and Chen Li, Sharad Mehrotra: “Executing SQL over encrypted data in the database-service-provider model”, SIGMOD Conference 2002, pp. 216-227

[23] Dirk Düllmann, Wolfgang Hoschek, Francisco Javier Jaén-Martínez, Ben Segal, Heinz Stockinger, Kurt Stockinger, and Asad Samar: “Models for Replica Synchronisation and Consistency in a Data Grid”, HPDC 2001, pp. 67-75

 ISBN: 978-1-61208-106-9

The Limitation of MapReduce: A Probing Case and a Lightweight Solution

Zhiqiang Ma Lin Gu
The Department of Computer Science and Engineering
The Hong Kong University of Science and Technology
Kowloon, Hong Kong
Email: {zma,lingu}@cse.ust.hk

***Abstract*—MapReduce is arguably the most successful parallelization framework especially for processing large data sets in datacenters comprising commodity computers. However, difficulties are observed in porting sophisticated applications to MapReduce, albeit the existence of numerous parallelization opportunities. Intrinsically, the MapReduce design allows a program to scale up to handle extremely large data sets, but constrains a program's ability to process smaller data items and exploit variable-degrees of parallelization opportunities which are likely to be the common case in general application. In this paper, we analyze the limitations of MapReduce and present the design and implementation of a new lightweight parallelization framework, MRlite. MRlite can efficiently process moderate-size data with dependences among numerous computational steps. In the mean time, the parallelization on each step emulates the MapReduce model. Hence, the MRlite framework can also scale up for large data sets if massive parallelism with minimal dependence exists. MRlite can significantly improve the flexibility and parallel execution performance for a number of typical programs. Our evaluation shows that MRlite is one order of magnitude faster than Hadoop on problems that MapReduce has difficulty in handling.**

***Keywords*-Distributed computing; Parallel architectures**

I. INTRODUCTION

MapReduce [1] is arguably the most successful parallelization framework used in datacenters comprising commodity computers [2]. The open-source variant of MapReduce, Hadoop [3], has seen active development activities and increasing adoption. Many cloud computing services provide MapReduce functions [4], and the research community uses MapReduce and Hadoop to solve data-intensive problems in bioinformatics, computational finance, chemistry, and environmental science [5][6][7][8].

On the other hand, the MapReduce model has its discontents. DeWitt et al. argues that MapReduce is much less sophisticated or efficient than parallel database query systems [9]. It is pointed out that the MapReduce model imposes too strong assumptions on the dependence relation among data, and the correctness often depends on the commutativity, associativity, and other properties of the operations [10]. Others point out that the unreliable communication model and retry mechanisms are far from being satisfactory, and the master node can easily become a single point of failure. The performance study on MapReduce-based algorithms exhibits mixed results [5]. Finally, the recently granted MapReduce patent raises question on the long-term viability of using this parallelization mechanism in open environments [11].

We argue, however, that the facts and observations above do not reveal the real limitation of the MapReduce technology–they are either not significant enough to taint the technical merits of MapReduce, or not technical issues at all. In addition to the capability of exploring massive parallelism, the MapReduce framework has its generality to make it attractive to a wide class of analytics applications. Otherwise, it would not have been used for many years as a fundamental piece of software in the Google architecture, which is a complex system solving many challenging problems [2].

The intrinsic limitation of MapReduce is, in fact, the "one-way scalability" of its design. The design allows a program to scale up to process very large data sets, but constrains a program's ability to process smaller data items. The one-way scalable design reflects assumptions made in a design context where large data sets were the dominating challenges, and affects several important design choices in the MapReduce framework.

While the one-way scalability was a legitimate choice when MapReduce was initially designed, it introduces severe difficulty in extending this programming framework to more general computation. It has become imperative to design a new parallelization framework that is not only scalable but also flexible and generally applicable as cloud computing evolves to cover more dynamic, interactive, and semantic-rich applications, such as multiple-user collaborative applications [12], scientific computing, development tools, and commercial applications [13].

In this paper, we use a specific case to probe the limitation of MapReduce, and design a new lightweight parallelization framework to mitigate the one-way scalability problem and improve the system performance. The probing case is a distributed compilation tool, and the new parallelization framework is called "MRlite", which can efficiently scale down to process moderate-size data. Our evaluations on MRlite show that it is more than 12 times faster than Hadoop in the distributed compiling workload.

The rest of this paper is organized as follows. Section II discusses related work. Section III describes the probing case and our evaluation on it. Section IV presents the design of

 ISBN: 978-1-61208-106-9

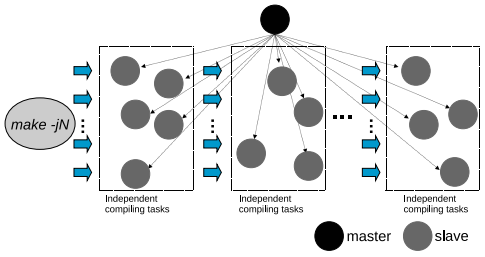


Figure 1. The architecture of mrcc

MRlite. Section V describes the prototype and the evaluation result of MRlite. We give a brieve conclusion in Section VI.

II. Related work

MapReduce is initially designed and implemented by Google for processing and generating large data sets [1]. Solutions to a wide class of real world problems can be expressed in this model. MRlite subsumes the parallel execution capability of MapReduce so that the problems handled by MapReduce can also be solved by MRlite. In addition, MRlite can handle workloads that MapReduce cnanot efficiently process.

The open-source variant of MapReduce, Hadoop [3], as well as its underlying data persistency layer, HDFS [14], which is loosely modelled after GFS [15], has seen active development and increasing adoption. Hadoop also has the "one-way scalability" limitation in its design. Different design choices are made in MRlite, which mitigate the "one-way scalability" problem.

Dryad is another distributed execution engine which allows an application to specify an arbitrary "communication DAG (directed acyclic graph)" [16]. Dryad also allows one vertex in the graph to consume multiple inputs and generate multiple outputs. DryadLINQ compiles the LINQ (a set of .NET constructs for queries) [17] programs into a distributed Dryad execution plan which can be executed directly on Dryad [10]. MRlite does not use the DAG based approach.

The limitation of MapReduce is also manifested in problems with large data sets. Chen et al. points out that it is tricky to achieve high performance for programs using Mapreduce, although implementing a MapReduce program is easy [18]. MRlite's programming interface and lightweight design help developers explore more potential parallelization opportunities in solving a wider range of problems.

III. A case study

To probe the limitation of the MapReduce framework, we design mrcc [19], a distributed compilation system, and examine its performance and overhead. MapReduce is not designed for the compilation workload which contains moderate-size data with complex dependency. We choose the compilation workload to probe the limitation of MapReduce. Meanwhile, a large class of applications share the features of compilation workload, such as variable-size data and dependency among them, and variable degree of parallelization at different algorithmic steps.

mrcc consists of one master node that controls the compilation job and many slave nodes that handle the compilation tasks as shown in Figure 1.

When one project is compiled on the master node, *make* builds the dependency tree for this project, and invokes multiple mrcc program instances to compile multiple source files in parallel. Hence, the parallelization are leveraged by *make* invoking multiple concurrent mrcc instances. Each mrcc instance runs one compilation task on a slave node. A slave node is one of the worker machines that receive map/reduce tasks from MapReduce master.

When conducting remote compilation on a slave node, mrcc preprocesses the source file, places a batch of preprocessed source files into a network file system used by the framework, then starts a compilation job on MapReduce. The map operation of this MapReduce job is done by a program called "mrcc-map". Running on the slave node, mrcc-map first retrieves the source file from the network file system, then calls the compiler locally to process the source file on the slave. After the compilation finishes, mrcc-map places the object file which is the result of the compilation back into the network file system. After the mrcc-map task is finished, mrcc on the master node retrieves the object file from the network file system and places it into the master node's local file system. After one batch of files are compiled, *make* continues to release more files to be compiled that depend on the completed ones.

A. Implementation

The mrcc compilation system consists of two core parts: the main program mrcc which runs on the master node and mrcc-map which runs on the slave nodes. mrcc is an open-source project under the GNU General Public License, version 2. The source files of mrcc can be downloaded from [19]. The work flow of the mrcc program is shown in Figure 2.

mrcc forks the preprocessor process after scanning the compiler arguments. The preprocessor inserts the header file(s) into the source file so that the remote nodes can assume a much simpler execution environment. mrcc then places the preprocessed file into a network file system and conducts remote compilation. When running mrcc on Hadoop, we use Hadoop Streaming [20] which can run MapReduce jobs with any executable or script to perform the map or reduce operation. mrcc submits the job to Hadoop and mrcc-map on the slave node is invoked by Hadoop to perform the map operation.

mrcc-map is implemented as a program residing in local directories of all the slave nodes because the mrcc-map program does not change during the process of compiling

 ISBN: 978-1-61208-106-9

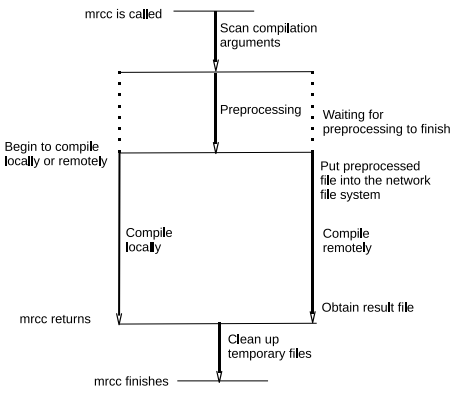


Figure 2. The work flow of mrcc

Table I
NODE CONFIGURATION

	CPU	Memory (GB)	Number
mrcc master	4	2	1
Slaves	2	2	10
Hadoop or MRlite master	2	2	1
NFS server	2	14	1

one project. This also makes it "easier" for the MapReduce framework to handle the compilation tasks, and, hence, the performance penalty we observe shall reflect more accurately the intrinsic limitations of the methodology.

mrcc-map first parses its arguments to obtain the source file name on the network file system and the compilation arguments. mrcc-map then retrieve the preprocessed file from the network file system. After that, mrcc-map calls the local *gcc* compiler and passes the compilation arguments to it. When *gcc* exits with a successful return value, mrcc-map places the object file into the network file system and returns immediately.

Upon the completion of all mrcc-map tasks, the Hadoop Streaming job returns. mrcc obtains the object files from the network file system and returns.

B. Performance of mrcc on Hadoop

We set up an experiment platform that consists of Xen virtual machines. We isolate these virtual machines by assigning each virtual machine except the mrcc master a physical CPU core which contains two CPUs. The configuration of the slave nodes are identical. Details of the node configuration are listed in Table I. The Hadoop master node and three slave nodes reside on one physical machine while the other seven slave nodes is on top of another physical machine. The mrcc master node is on top of the third physical machine. The physical machines are connected by a Netgear JG5516 1Gbps switch. The bandwidth between two virtual machines on top of one physical machine is also 1Gbps.

Table II
TIME FOR COMPILING PROJECTS USING GCC ON ONE NODE AND MRCC ON HADOOP

Project	Time for gcc	Time for mrcc/Hadoop
Linux	48m56.2s	150m44.4s
ImageMagick	5m12.1s	10m52.7s
Xen tools	2m7.6s	23m38.9s

We use mrcc to compile the Linux kernel 2.6.31.6, ImageMagick 6.6.3-8, and Xen tools 4.0.0 on Hadoop to examine the performance of Hadoop when it processes jobs in which complex dependencies exist between tasks while the input data files are also dynamically generated. The Hadoop version is 0.20.2 [3].

Table II shows the performance data. The compilation time using mrcc on 10 nodes is at least twice as long as that on one node (sequential compilation). Further investigation reveals that Hadoop takes more than thirty seconds to complete one compilation task. We also measures that Hadoop takes more than 20 seconds to finish one "null" job even though the job does not do any work. Storing or retrieving one file on the network file system takes at least 2 seconds while compiling one file on one node usually takes less than 2 seconds. While such overheads are acceptable in the special class of applications where relatively simple processing logic is applied to a large number of independent data, the prohibitive tasking and data transportation cost limits the applicability of MapReduce/Hadoop in more general workloads.

IV. DESIGN

The experimental results in Section III-B show that the current design and implementation of the MapReduce framework cannot provide the flexibility and efficiency required by programs with numerous parallelizable steps, instead of one massive parallelizable step. Hence, the current MapReduce framework does not work effectively for a large class of applications with not only sizable data but also non-trivial application logic. Representing the "common" case in scientific and business computing, such applications require the programming framework to efficiently handle variable-size data, support data dependence, and harness variable degrees of parallelization with controlled latency.

To overcome the limitation of MapReduce, we have designed and implemented MRlite, a lightweight parallelization framework that optimizes for not only massive parallelism, but also low latency to provide a more general and flexible parallel execution capability in cloud computing environments. The data in a complex computing system are often dynamically generated, thus introducing dependence among data. In fact, "data with dependence" shall be considered the common case in general applications. Such

 ISBN: 978-1-61208-106-9

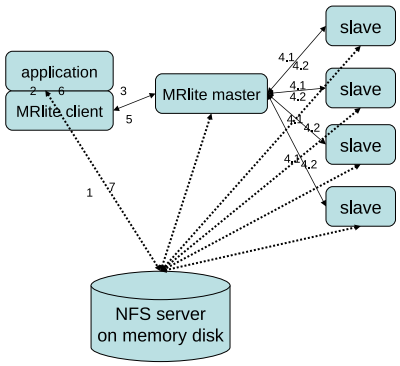


Figure 3. The architecture of MRlite

dependence naturally divides the processing into numerous steps to be taken in order, but each step may have sufficient parallelism to be exploited. The key is, consequently, to significantly reduce the overhead in data transportation and task management so that most of the parallelizable steps can invoke the parallelization mechanism, with the performance gain from parallel execution outweighing the latency induced by the overheads.

A. Architecture

The MRlite system consists of four parts as shown in Figure 3: the MRlite master, MRlite slaves, the NFS server in memory, and the MRlite client.

The MRlite master controls the parallel execution of tasks. It accepts jobs from the MRlite client, and distributes the tasks of the jobs to MRlite slaves. The MRlite slaves accept and execute the tasks from the MRlite master. The MRlite client is a library that can be linked to ther user application. The MRlite client accepts the application's parallelization requests, and submits the job to the MRlite master. MRlite includes an NFS server whose files are stored in one particpating node's memory to profide a file system abstraction. The NFS file system is mounted on the MRlite master node, all the MRlite slave nodes, and the node on top of which the user application runs.

B. Latency optimization

The MRlite master cooperates with the MRlite client to minimize overhead and perform low-latency operations. In addition, the MRlite framework includes a timing control feature in its design as part of the low-latency execution mechanism. The application can estimate a timeout limit for the job and provide the timeout limit when it sends parallelization request for one job to the MRlite client. According to the timeout limit for the whole job, the MRlite master provides a suggested timeout value for the tasks to be executed on the slaves. In the current design, the timeout value is specified by the programmer. In the future work, we will extend this to a more flexible mechanism. After receiving the task command from the master, the slave tries its best to complete the task during the time slot specified by the timeout value. The timing enforcement also take care of reliability through retries. If one task times out, the master treats it as a failed one and may retry that task on another slave or just report the failure to the master. Similarly, a timed-out job may be treated as a failed one by the MRlite client, and the client may retry the job for a certain number of times or report the failure to the application according to that job's configuration. The application logic ultimately decides how to proceed when a job fails.

Besides the execution time enforcement, the MRlite master submits tasks to slaves without sophisticated queueing to maximize the possibility of finishing the job within the timeout limit. As there are dependences between jobs and among map and reduce tasks, the master submits the tasks as soon as the dependence is resolved.

The latency caused by the run-time overhead in each step may become critical when processing moderate data sets though it may not be a concern for processing a huge amount of data. Unlike the Hadoop design, MRlite uses a run-time daemon and thread pools to support the operations of the master and the slaves. This design reduces the cost of creating a process every time a job request or task request is issued. As the multi-thread and multi-core technology is widely available in modern computing platforms, we believe it is a pleasantly acceptable cost to dedicate one thread for each task on a slave and one thread for managing a job on the master.

Data transportation is an important aspect in distributed computing. In our lightweight parallelization framework, it is convenient to provide a distributed file system to store data, but we only store intermediate data files and the run-time data in the network file system. The design choice on the network file system implementation must balance the performance and usability. MRlite includes an NFS server, which runs on one participating node, to provide a file system abstraction, and mount the NFS file system on the MRlite master, all MRlite slaves, and the MRlite clients. The NFS server rides on a *tmpfs* file system [21], a virtual memory file system in Linux, so that the I/O speed of the NFS directory is as fast as operations in memory. This design choice reflects the observation that modern gigabit and 10 gigabit NICs provide comparable throughput bewteen networked computers to the I/O bandwidth between memory and hard drives. To maximize the I/O speed of the network file system, we do not duplicate the intermediate files as some distributed file systems do [15].

In the current design of MRlite, data persistency is supposed to be provided by a separate layer of data storage. The software based reliability through multiple-way replication is not included in MRlite since these are not the focus

 ISBN: 978-1-61208-106-9

of this work, and solutions that provide these features already exist [14][15]. Replication can potentially increase the serving bandwith of read operations, at the cost of first increasing the cost of writing operations. Both GFS and HDFS employs 3-way replication [14][15] to improve concurrency and reliability. In our experiments, it has not been observed that the lack of 3-way serving bandwidth limits the parallelization capability or the overall application-level performance when the network bandwidth is sufficient. Nevertheless, the MRlite architecture does not prohibit the addition of a replicated data storage, given that intermediate data files are still stored on and served from the low-latency network file system.

C. Programming interface

The MRlite client is designed as a library that can be compiled and linked to the user application. It provides a simple API so that the application developers can use the parallel computing capacity by simply calling a function. The developer can define the map program, the reduce program, how many map tasks and reduce tasks should be invoked, the input data directory/file, the output data directory/file and the time out value for the job. The MRlite client parses the application's parallelization requests, and submits the job to the MRlite master with the options specified in the API.

V. Prototype and Evaluation

We have prototyped the MRlite parallelization service, and implemented mrcc on MRlite. In this section, we discuss the implementation details of MRlite, and report the evaluation results of mrcc on MRlite.

A. Implementation

The MRlite jobs are represented as sets of native Linux applications written in any programming language of choice. After each job is split into numerous tasks, each task is executed as a Linux program by one of the MRlite slaves.

At each parallelizable step, the MRlite framework dispatches a group of concurrent tasks, emulating the MapReduce model inspired by list primitives in Lisp. The tasks executed on MRlite slaves are defined as map and reduce tasks, with reduce tasks aggregating the intermediate results generated by the map tasks. It worths noting that we do not restrict map and reduce tasks to use key/value pairs. The MRlite slaves monitor the tasks' execution and report the execution's status and return values to the MRlite master.

There are 7 steps during the process of executing one job as shown in Figure 3:

1) The application places input data files to the input NFS directory.
2) The application submits the MapReduce job to the MRlite client.

Table III
Comparison of speedups of mrcc on Hadoop and MRlite

	Speedup on Hadoop	Speedup on MRlite	MRlite vs. Hadoop
Linux	0.32	5.8	17.9
ImageMagick	0.48	6.2	13.0
Xen tools	0.09	2.0	22.0

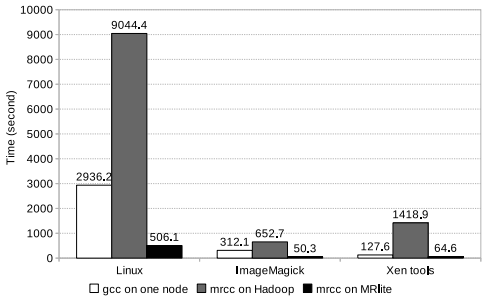


Figure 4. Execution time for compiling projects

3) MRlite client accepts the job, and submits it to the MRlite master.
4) The MRlite master submits tasks of the job to slaves (4.1), and waits for slaves to respond (4.2). Perform steps 4.1 and 4.2 for all tasks.
5) The MRlite master sends a success or fail message back to MRlite client.
6) MRlite client returns to the application with a return value that represents the result.
7) The application retrieves the output data files from the output NFS directory.

B. Evaluation

Because MRlite is a general parallelization framework, we can port the mrcc program to MRlite, and compare the performance with mrcc on Hadoop using the workloads as described in Section III-B with complex dependencies existing among tasks and some input data files dynamically generated. This evaluation examine MRlite's ability to scale "down" to handle moderate-size data, mixed sequential and parallel work flow, and latency-aware tasks.

Most of the components in the mrcc/MRlite implementation are identical to those in the Hadoop based implementation except the parts invoking programming interface. The configurations of the nodes and the client nodes are listed in Table I. The MRlite platform has the same number and hardware configuration of slave nodes as Hadoop in Section III-B.

Figure 4 shows the evaluation results. The compilation of the three projects using mrcc on MRlite is much faster than compilation on one node, with a speedup of at least 2 and the best speedup reaches 6. Table III lists the speedup mrcc achieves on Hadoop and MRlite, and shows that the

 ISBN: 978-1-61208-106-9

average speedup of MRlite is more than 12 times better than that of Hadoop. This comparison shows that the MRlite is more effective than MapReduce for workloads with numerous computational steps where each step may have parallelization opportunities. From the result we also find that a project that is easier (a higher speedup) for mrcc on Hadoop to compile is easier for mrcc on MRlite. When compiling the "easier" project such as ImageMagick, mrcc on MRlite can achieve better speedup than mrcc on Hadoop.

The evaluation shows that MRlite is one order of magnitude faster than Hadoop on problems that MapReduce has difficulty in handling. The MRlite framework can still handle the massive parallelism with simple dependency as MapReduce does. MRlite shows that we can implement a flexible and efficient parallelization framework which programs can easily invoke to handle both large and small data sets.

VI. Conclusion

In this paper, we use the distributed compilation case to probe the limitation of MapReduce. The probing case shows that the overhead of Hadoop is too high for mrcc and the performance of mrcc on Hadoop is far from satisfactory.

We design MRlite, a new lightweight parallelization framework, to mitigate the one-way scalability problem and improve the system performance. The evaluation result shows that MRlite can efficiently process moderate-size data and handle data dependence. The MRlite framework can still handle the massive parallelism with simple dependency. The design significantly improves the parallel execution performance for general applications.

VII. Acknowledgment

This work is supported in part by HKUST research grants REC09/10.EG06 and SBI08/09.EG03.

References

[1] J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters." in *the 6th Conference on Symposium on Operating Systems Design & Implementation*, vol. 6, San Francisco, CA, 2004, pp. 137–150.

[2] L. Barroso, J. Dean, and U. Hoelzle, "Web search for a planet: The Google cluster architecture," *IEEE Micro*, vol. 23, no. 2, pp. 22–28, 2003.

[3] "Apache Hadoop," *http://hadoop.apache.org/*, [last access: 9/2, 2010].

[4] "Amazon Elastic Compute Cloud – EC2," *http://aws.amazon.com/ec2/*, [last access: 9/2, 2010].

[5] C.-T. Chu, S. K. Kim, Y.-A. Lin, Y. Yu, G. Bradski, A. Y. Ng, and K. Olukotun, "Map-Reduce for machine learning on multicore," in *Proc. of NIPS'07*, 2007, pp. 281–288.

[6] M. C. Schatz, "CloudBurst: highly sensitive read mapping with MapReduce," *Bioinformatics*, vol. 25, pp. 1363–1369, 2009.

[7] J. Ekanayake, S. Pallickara, and G. Fox, "MapReduce for data intensive scientific analysis," in *Fourth IEEE International Conference on eScience*, 2008, pp. 277–284.

[8] Y. Chen, D. Pavlov, and J. F. Canny, "Large-scale behavioral targeting," in *KDD '09: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2009, pp. 209–218.

[9] D. DeWitt and M. Stonebraker, "MapReduce: a major step backwards," *http://databasecolumn.vertica.com/database-innovation/mapreduce-a-major-step-backwards/*, [last access: 9/2, 2010].

[10] Y. Yu, M. Isard, D. Fetterly, M. Budiu, Ú. Erlingsson, P. K. Gunda, and J. Currey, "DryadLINQ: A system for general-purpose distributed data-parallel computing using a high-level language," in *OSDI*, 2008, pp. 1–14.

[11] D. Jeffrey and G. Sanjay, "System and method for efficient large-scale data processing," U.S. Patent 7,650,331, 2010.

[12] "Google Docs," *http://docs.google.com*, [last access: 9/2, 2010].

[13] "Salesforce.com," *http://www.salesforce.com*, [last access: 9/2, 2010].

[14] "Hadoop Distributed File System," *http://hadoop.apache.org/hdfs/*, [last access: 9/2, 2010].

[15] S. Ghemawat, H. Gobioff, and S.-T. Leung, "The Google file system," in *Proc. of the 9th ACM Symposium on Operating Systems Principles (SOSP'03)*, 2003, pp. 29–43.

[16] M. Isard, M. Budiu, Y. Yu, A. Birrell, and D. Fetterly, "Dryad: distributed data-parallel programs from sequential building blocks," in *EuroSys '07: Proceedings of the 2nd ACM SIGOPS/EuroSys European Conference on Computer Systems 2007*, 2007, pp. 59–72.

[17] D. Box and A. Hejlsberg, "LINQ: .NET language-integrated query," *http://msdn.microsoft.com/library/bb308959.aspx*, [last access: 9/2, 2010].

[18] S. Chen and S. W. Schlosser, "Map-Reduce meets wider varieties of applications," Intel Research Pittsburgh, Tech. Rep. IRP-TR-08-05, May 2008.

[19] Z. Ma, "mrcc," *http://www.cse.ust.hk/~zma/proj/mrcc.html*, [last access: 9/2, 2010].

[20] "Hadoop Streaming," *http://hadoop.apache.org/common/docs/r0.20.2/streaming.html*, [last access: 9/2, 2010].

[21] P. Snyder, "tmpfs: A virtual memory file system," in *In Proceedings of the Autumn 1990 European UNIX Users Group Conference*, 1990, pp. 241–248.

 ISBN: 978-1-61208-106-9

The Business Model of Cloud Storage

Jincai Chen, Minghui Lai, Yangfeng Huang, Kun Yang, Gongye Zhou
Huazhong University of Science and Technology - Wuhan, China
jcchen1@sohu.com ; firefly727@qq.com ; huyfaeng@163.com ; hustyk@163.com ; zhougongye@126.com

***Abstract*—The traditional service model of cloud storage is that the service providers supply both the storage capacities and data storage services through the Internet to the clients. The obvious disadvantage of this model is that the data of the same customer stored in multiple cloud storage providers can not interact with each other. This paper adds a new layer to this model. The new model consists of StaS (Storage as a Service) User, StaS Provider and Cloud Storage Provider, which can solve the above problem. It also possesses some other advantages, such as Cloud Storage Providers need not care about the market and how customers use their services, different customers can use the cloud storage services in different security levels as they need. This paper also discusses the functions of different modules at StaS Provider layer and analyzes two architectures of cloud storage at Cloud Storage Provider layer.**

Keywords-cloud storage; StaS; business model

I. INTRODUCTION

Nowadays the growth of data is just like flood overflowing. Enterprises, governments, the non-profit organizations and consumers are facing more and more stern challenges, such as data storage, data management, data protection, data excavation and so on (especially in the data storage). The traditional storage devices and storage methods have no ability to deal with such a huge data quantity. The birth of cloud storage can alleviate the pressure of the storage of mass data.

We can say that the birth of cloud computing have made cloud storage. Cloud storage develops very quickly on the basis of cloud computing. We all know that cloud computing is provided and researched by some well-known companies. So the technology of cloud storage mainly take some cloud storage products of some well-known companies as representative, such as Amazon's S3 (Simple Storage Service), Nirvanix's SDN, EMC's Atmos, IBM's Blue Cloud and Microsoft's Live Mesh [1-3].

These cloud storage products have their own advantages and disadvantages. Customers can choose different cloud storage products according to their needs. With the appearance of various cloud storage products, a serious problem occurred. Customers can not use multiple cloud storage services provided by different cloud storage providers because of different pricing standards, different access interfaces and different storage forms of data at the same time. SNIA has found this problem, so it began to do something on the relevant standard of cloud storage in 2009 [4]. SNIA presents a standard about Cloud Data Management Interface (CDMI) in September 2009 [4]. The version 1.0 of CDMI was released in 2010. The main content of this standard is the data access mode, data organization and management, classification and metadata management and data security in the cloud storage system. This standard has an undoubtedly huge promotion on the development of cloud storage and the compatibility of different cloud storage products.

As enterprises have an enthusiasm to cloud storage, more and more researchers do academic research on cloud storage. The main research content is as follows: the management of data in cloud [5-7], the architecture of cloud storage [8-10], the security of cloud storage [11], the migration of data in cloud system [12] and so on.

The rest of the paper is arranged as follows. Section 2 simply introduces the business model of cloud storage and its advantages. Section 3 introduces the details of StaS Providers Layer and the function of each module. Section 4 compares two architectures of Cloud Storage Providers, P2P architecture and Master-Slave architecture, in control way, fault tolerant and load balancing. Section 5 makes a conclusion.

II. BUSINESS MODEL OF CLOUD STORAGE

Patterson has provided the Business Model of Cloud Computing [13], which consists of SaaS User, SaaS Provider and Cloud Provider as follows:

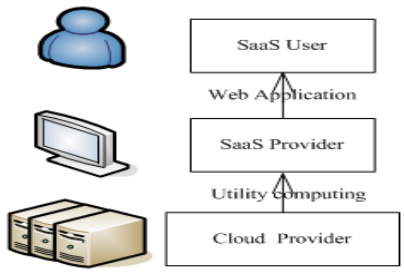


Figure 1. The Business Model of Cloud Computing [13]

So SaaS (Software as a Service) Provider can provide cloud computing service without any expensive hardware storage devices, and they only try their best to exploit the parallel software, whereas Cloud Provider only needs to manage their own cloud computing system and try their best to enable the cloud storage system to be more reliable, available, secure and effective. The advantages of this model are that Cloud Provider has no need to take care about market and customers. So cloud computing will develop more quickly and spread more widely.

In fact, the business model of cloud computing can be also applied to cloud storage, which consists of StaS User,

 ISBN: 978-1-61208-106-9

StaS Provider and Cloud Storage Provider; nevertheless, the StaS here is Storage as a service (for distinguishing from Software as a Service, note that all StaS of this paper behind is referred to Storage as a Service.) and the Figure is as follows:

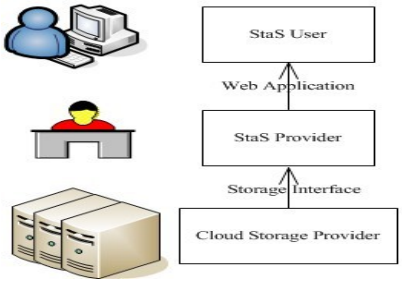


Figure 2. The Business Model of Cloud Storage

In this Business Model of Cloud Storage, Cloud Storage Provider only needs to manage the cloud storage system effectively, such as the deployment, the architecture, fault-tolerant and the interface which is provided to StaS Provider, etc. StaS Provider needs not to own enormous storage devices. However, it owns infinite storage capacity which can be assigned to StaS User. StaS Provider should have the functions as follows: managing the information of StaS User, purchasing storage space from different cloud storage providers, mapping this physical space to a logical space for StaS User through Virtual technology, providing different customers with different security levels.

The advantages of this business model of cloud storage include the following:

1. StaS User can interact with their data which is stored in multiple cloud storage systems provided by different cloud storage companies through promoting data interoperability to the StaS Provider layer. Because StaS Provider can call the corresponding interfaces functions of the cloud storage system, multiple physical spaces of different cloud storage providers can be added to a StaS User's logic space, which is just like mounting multiple disks.

2. StaS Provider can provide different StaS Users with different security levels. The higher security level is, the higher the price is. The reason why the previous cloud storage can not occupy the market is that cloud storage can only provide high security and the price is expensive. Many customers, such as general customers, do not need such a high security level. So they lose a lot of customers.

3. Owing to StaS Providers can concentrate storage space belonged to multiple Cloud Storage Providers, StaS Providers can own as much storage capacity as they want.

4. It is not necessary for Cloud Storage Providers to pay much attention to market and customers. Cloud Storage Provider in the traditional model maybe have to care about one million customers, whereas in this model maybe only care about one or several customers who are StaS Providers. They only need to pay all their attention to the design and management of cloud storage system (such as the arrangement, the architecture, the secure mechanism and so on).

III. STAS PROVIDER

The details of this business cloud storage model are as follows:

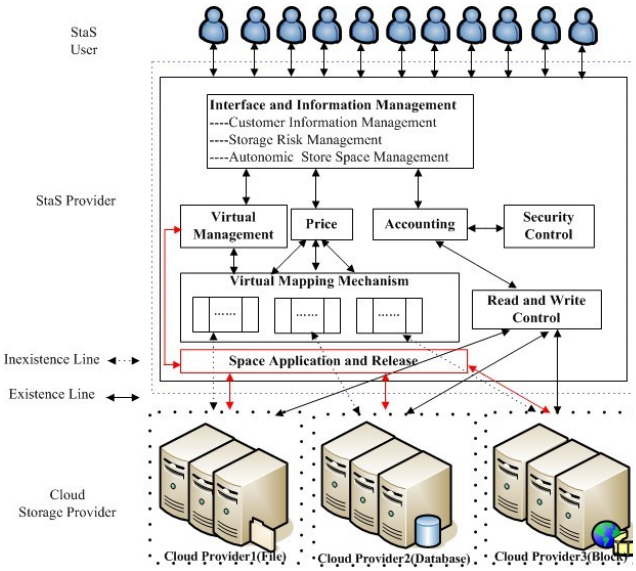


Figure 3. The Detail Business Cloud storage Model

From Figure 3 we can see that StaS Provider Layer own many modules. The functions of each module are as follows:

Interface and Information Management(IIM):This module provides interface for StaS User and manages the information of customers. Also it should monitor the risk of customers' application. Only in the none-risk case, the system will allow customers to apply for space. The last function of this module is the management of the logical store space.

*Accounting:*This module manage the status of customers` login. After a customer logs in, he/she should get the information data from the module of Interface and Information Management and then Accounting will make a map between customer's login and the information of his data (the information of data is equivalent to the customer's logical storage space).

*Virtual Management(VM):*This module manages and monitors the below logical storage space. When StaS Provider apply for storage space from Cloud Storage Provider through the module of Space Apply and Release, VM need to map the newly applied physical space to the logical space and manage it. When StaS User applies for some storage spaces, VM need to maintain available spaces for each customer. Of course, there is an assumption that Cloud Storage Provider can provide high availability (Even if there are server crashed in Cloud Storage Provider layer, there should be corresponding alternative server and the map with Virtual Mapping Mechanism can not change).

*Security Control(SC):*Different customers have different requirements for the security of their data. For example an enterprise demands a very high security level, whereas common customer does not need such a high security level. So StaS Provider can provide different security levels for

different customers through SC. When a customer logs in StaS Provider, he/she should choose the security level through SC.

*Price:*This module is used to calculate the cost of customers. The traditional model of cloud storage do not own the layer of StaS Provider, so customers can not interact with their data stored in multiple cloud storage providers because of the different pricing standards and the different access interfaces. But customers can do this in this model. Because this module can calculate the cost of customers according to corresponding cloud storage manufacturer's charging standard. So customers can use storage space of different manufacturers, which is just like mounting disk.

*Read and Write Control(RWC):*This module can let customers use multiple storage services provided by different cloud storage manufacturers. We all know that there are different interfaces and storage forms (such as file, database, block and so on) among different cloud storage services. RWC can shield these differences. When customers access their data, they only need to send the access demand, the logical space of data, and the corresponding manufacturer's name to RWC. Then RWC will call corresponding access interface.

*Space Application and Release(SAR):*This module is used to apply for or release space from the layer of Cloud Storage Provider. When applying to the new space, it should finish the additional logical space mapping with the above VM. When releasing the space, it should cut the logical space with VM.

*Virtual Mapping Mechanism(VMM):*This module only map physical space to logical space and the above VM take charge of the management of this logical space. If we do not care about other modules, we can abstract the system as Figure 4. We can see that the space of all physical storage devices is mapped to a huge logical storage space and customers apply for logical space from StaS Provider.

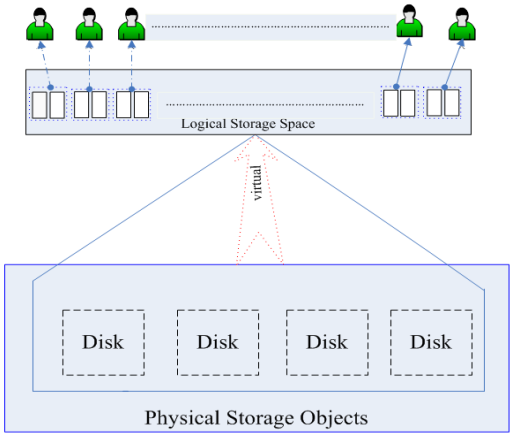


Figure 4. Abstract of Virtual Mapping

Now suppose that the total logical capacity is 1024TB and the average size of each customer purchase is 1 TB. So the number of customers StaS Provider can hold is 1024 at most. If there are any more new customers who want to purchase logical storage space, StaS Provider have to purchase more physical storage space from Cloud Storage Provider. Are there any methods which can tolerate that some customers continue to buy logical space from StaS Provider and by which StaS Provider have no need to purchase physical space from Cloud Storage Provider? Meanwhile we promise that each online customer's storage space is the same size as they have purchased.

In fact, there are two important statuses. Firstly, the possibility that all the customers are online is very small, or we can say that the number of online customers is just a certain amount of total customers. Secondly, the storage space that customers purchased will not be used up completely. For these two reasons, we can solve the above problems. We can recycle the unused storage space of the offline customers (Note: we just recycle it temporarily. If customers are online again, we will assign the space of the same size to them from other unused space) and allow online customers or new customers to purchase this space. If we do this, the system of the above example can hold more than 1024 customers.

For example, suppose that the average online rate is two-thirds and the average use rate of customers` space is two-thirds, then we can hold at least 110 customers to buy storage space of 1TB practically, according to

$$1024*(1-\tfrac{2}{3})*(1-\tfrac{2}{3})=113.78.$$

We just simply suppose the average online rate and the average use rate of space. If we want to do further research in this aspect, we can analyze the discipline of the customers (the average online rate or offline rate) and how much the average use rate of space is, we can hold the largest number of customers. Of course, we should pay attention to the number of spare capacity of the system and the present risks. We can create an optimization model with these parameters.

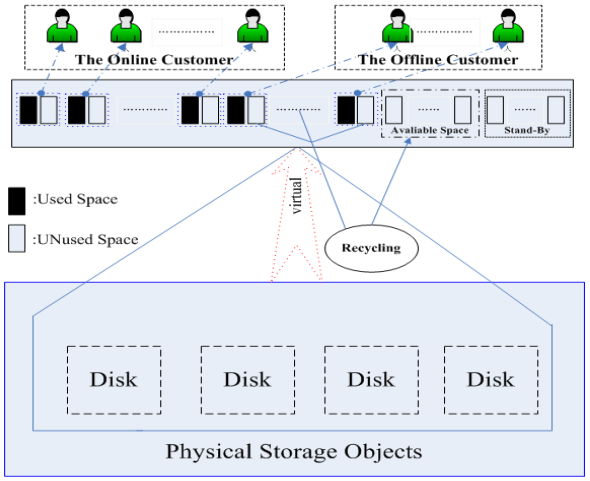


Figure 5. The Space Recycling Mechanism

If we take this mechanism, StaS Provider back up a certain space for emergency (the total space that online customers purchased is larger than the total space that StaS Provider can assign).The details are as shown above.

We can add the size of this emergency backup space to the above optimization model. But this mechanism has a high demand for the virtual technology and the management

 ISBN: 978-1-61208-106-9

of space is a little complex. Of course, this is a very good mechanism.

IV. CLOUD STORAGE PROVIDER

As the real physical storage devices provider, Cloud Storage Provider should provide StaS Provider layer with high availability of services. It not only provides reading and writing interfaces, but also needs to ensure the security of data and the availability. As it should try to improve the efficiency, there would involve a lot of technologies. Now we will discuss Cloud Storage Provider as follows: model, architecture and instance.

A. Model

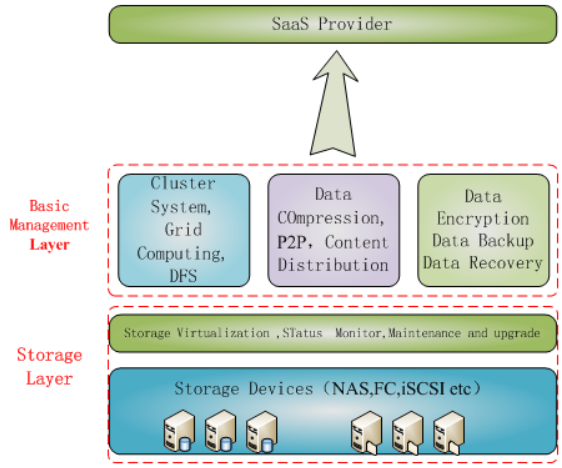


Figure 6. Cloud Storage Model

Compared with traditional storage systems, cloud storage is a complex system. It includes not only the hardware, but also the network equipment, the storage equipment, servers, applications, public access interfaces, access networks and so on. It provides data storage and business access services through the application software. Cloud Storage can be divided into two Layers: Basic Management Layer and Storage Layer.

1) Storage Layer

Storage Layer is the most basic part of the Cloud Storage Structure. Storage devices can be FC storage devices, also can be IP storage devices, such as NAS and iSCSI. Storage devices in the Cloud Storage Network are often located in different regions, and even different countries. The nodes can be linked together through the storage network and the storage network can be SAN, NAS, FC-SAN, etc.

Storage Device Management System is on the top of storage devices. It can mask the differences between various physical storage devices. So it can achieve storage devices logical, management virtualization, multi-link redundancy management, and hardware status monitoring and troubleshooting.

2) Basic Management Layer

This layer is the key part of the cloud storage system and is also difficult to realize. Basic Management Layer make multiple storage devices to work together, the external provision of service and provide better data access performance through the cluster, the distributed file system and the grid computing technologies.

Data encryption stored in the cloud will not allow unauthorized user access, while a variety of data backup and disaster recovery technology can guarantee that the data in the cloud storage will not be lost and the data will be safe and stable.

B. Architecture

Cloud storage network provides data backup, data migration and other operations through the internal network, and cloud storage services through the external network. Cloud storage system is a new storage system. In order to deal with immense data generated everyday. Generally speaking, when we build a cloud storage system, we should consider several aspects as follows:

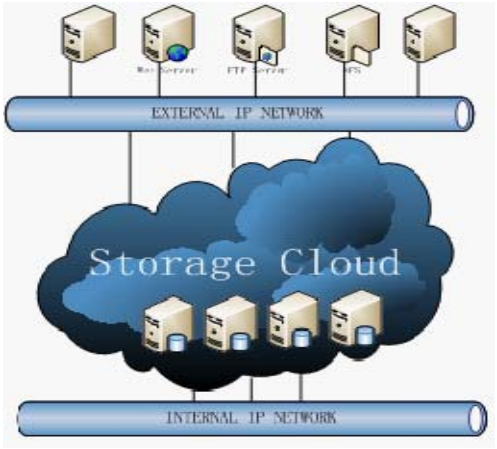


Figure 7. Cloud Storage Architecture

Expand the capacity: When increasing the capacity of the cloud storage system, we should be able to provide services continually, add the new storage nodes to the original storage pool automatically, and do not need too much duplication and complex configuration.

Reliability and Availability: Storage node failure is normal, not abnormal. When the storage node fails, ensure that we can provide cloud storage services continually, and the data in the cloud storage is not lost.

Management: Cloud storage networks include thousands of storage nodes, how to manage so many nodes effectively will become a key to how to build architecture successfully.

Costs: Because the cloud storage system has the same service interface with the tradition network storage system, it can be integrated into the existing storage system conveniently and do not require any structural change in the existing system. It can greatly reduce the cost of the deployment of the cloud storage system.

C. Cloud storage architecture instance

At present, there are two forms of typical cloud storage architecture, Master-Slave architecture and P2P architecture. Each company can select a model according to their characteristics. Each architecture has its own advantages and

 ISBN: 978-1-61208-106-9

disadvantages, we will discuss control way, fault tolerance and load balancing as follows.

1) Master-Slave architecture

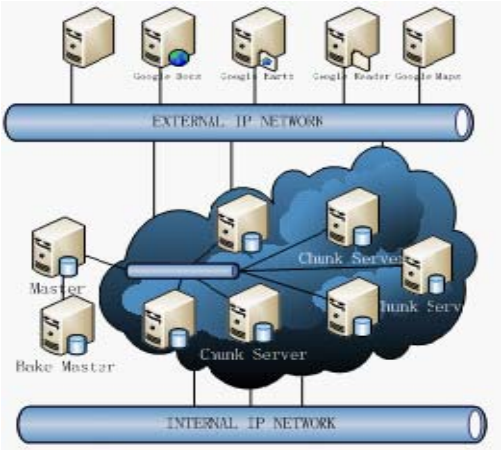


Figure 8. Master-Slave Architecture

a) Control way

This structure consists of a master server and many storage nodes; data storage format is a multi-dimensional map of the sparse structure. Data (including the index, log, and record data) is ultimately stored in the distributed file system. Data is assigned to each node by master server, which monitors the status of every storage node and storage load balancing between nodes. The client read the index file stored in the master server by the pre-reading and cache technology. This Master-Slave model has an obvious disadvantage. There is a single point of failure. In order to avoid the master became the bottleneck of system performance and reliability. We need remote backup for master. However, there is an obvious advantage of this architecture. The system can be easily controlled, easily maintained, easily added a host to it, and there are no data consistency problems.

b) Fault-tolerant

As the data is stored in many normal PC, the machine failure is normal, not abnormal. Usually, the data will have more than one copy. The copies are stored in different machines, different racks, and even different data centers. When one machine failed, the system can provide data service continually to ensure high availability.

If a data need N copies, of course, N can be configured by the user .Generally speaking, N=3. When the data is written to the storage node, according to relative algorithm, the system will write N-1 copies to other N-1 designated storage nodes. So it can ensure that every data has multiple copies stored in cloud storage system. The location information of the data and copies will be saved in the master server. When the clients accesse their data, at first, they need visit metadata stored in master server to get location information. Then according to the location information, the client can access the corresponding data storage nodes to read their data. So when the individual storage node fails, the system can still guarantee the integrity of the data, and client can still read their data. In addition, in order to recover data, when a machine fails, the master server gets the copy from other normal machine and transport the copy to the abnormal machine.

Since location information of all kind of data are stored in the master server. In order to ensure the high availability of data, there will be a metadata backup.

c) Load balancing

According to the load information migrate data, master server monitor the load information of every storage node.

2) P2P Architecture

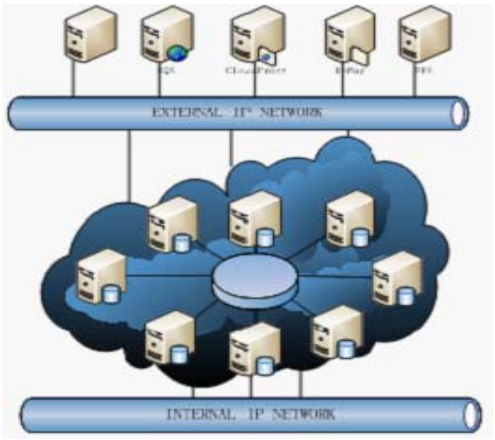


Figure 9. P2P Architecture

a) Control way

This architecture uses an improved DHT as its basic storage structure. The most important characteristic of this architecture is that the data are distributed on every storage node uniformly. Each storage node can communicate with each other .The data can be transported among the various nodes and detected if the data is in fault status. The advantage of this storage structure is that it has no single point of failure, and no master node control, but has self-management ability. The disadvantage is that there is data consistency problems, and inconvenient when adding the host to this system.

b) Fault-tolerant

The data stored in this structure uses redundant storage strategy as well as in the Master-Slave structure. We can use Dynamo's [14] strategy of redundant copies for reading and writing. It defines three parameters N, R, W. N represents the number of copies that each records. W represents the number of copies for each process of writing. R represents the number of copies for each reading. As long as R+W>N, we can read the latest copies. The number of R and W can be configured by us. We can update the lower version of data while we read the higher version of data. Thus we can handle the data consistency problem easily and adjust the demand of the high reading and high writing flexibly.

In addition, we can use Cassandra's [15] fault detection and recovery strategies, which determine the survival state of every storage node not by a BOOL value. Instead the failure detection module emits a value which represents a

 ISBN: 978-1-61208-106-9

suspicion level for each of monitored nodes. This value is defined as Φ [16]. According to the size of this value, the system detects mal functions. The performances of Accrual failure detector in accuracy and speed are very good; they can be adjusted to different network environment and server load environment.

c) Load balancing

Because the structure of DHT uses a way that the data is distributed evenly across the different nodes, there are no hot issues. Access pressure of every storage node is balanced. We can also use virtual node like Dynamo. When the machine has high performances, it can be configured with more virtual nodes. While the machine has low performances, it can be configured with less virtual nodes. The number of virtual node configured in each machine is determined by the machine's performance. Heterogeneous machines can be easily managed, and the load of each machine is also more balanced. The number of virtual nodes should be far greater than the number of physical machine. Many virtual nodes correspond to a physical machine. When we add a new machine to the system, we need not re-HASH, but just need to move some virtual nodes to this new machine. So it significantly reduces the amount of data moved.

V. CONCLUSION AND FUTURE WORK

This paper adds the StaS Provider layer to the business model of cloud storage. This model can solve the problem that customers can not interact with their data stored in multiple cloud storage providers because of the different pricing standards and the different access interfaces. The key research contributions of this work include:

- We add the StaS Provider layer to the traditional model of cloud storage. So the new model of cloud storage is consisted of StaS User, StaS Provider and Cloud Storage Provider and Advantages of this model is discussed.
- The functions of the different modules at StaS Provider layer (such as virtual management. security control and so on) are discussed and a virtual mechanism which can hold more capacity for customers by recovering the unused space of the offline customers is provided.
- The model of the Cloud Storage Provider layer is provided and some relevant functions which should be provided by it are discussed. Two practical architectures are provided, one kind is the architecture of Master-Slave, and another kind is the architecture of P2P. We have compared them in the following three aspects: control mode, fault-tolerant and load balancing.

The prototype system of the business model of cloud storage mentioned above has been building. Our next work is to test and analyze the performance of the system**,** practically and theoretically

ACKNOWLEDGMENTS

The authors thank Mr. Ming Chen and Ms. Ning Wang for their helpful discussions. This work was supported by the National Natural Science Foundation of China (Grant No. 60773189), the National Basic Research Program of China (No. 2004CB318201), and the Program for Changjiang Scholars and Innovative Research Team in University (No. IRT-0725).

REFERENCES

[1] http://aws.amazon.com/s3/. 06/15/2010.

[2] http://www.nirvanix.com/. 06/15/2010.

[3] http://www.emc.com/products/detail/software/atmos.htm. 06/15/2010.

[4] SNIA, "Cloud Data Management Interface," Version1.0. http://www.sina.org/tech_activities/standards/curr_standards/cdmi/. 06/15/2010.

[5] Raghu Ramakrishnam, "Data Management in the Cloud," 2009 IEEE International Conference on Data Engineering. pp. 5-5, April, 2009.

[6] ZHAN Ying and SUN Yong, "Cloud Storage Management Technology," 2009 Second International Conference on Information and Computing Science. pp. 309-311, May 21-22, 2009.

[7] Beng Chin Ooi, "Cloud Data Management Systems: Opportunities and Challenges," skg, pp. 2-2, 2009 Fifth International Conference on Semantics, Knowledge and Grid, 2009.

[8] Wenying Zeng, Yuelong Zhao, Kairi Ou, and Wei Song, "Research on Cloud Storage Architecture and Key Technologies," 2009 International Conference on Computer Sciences and Convergence Information Technology. pp. 1044-1048. 2009.

[9] Ke Xu, Meina Song, Xiaoqi Zhang, and Junde Song, "A Cloud Computing Platform Based on P2P," 2009 IEEE International Symposium on IT in Medicine & Education (TIME2009). pp. 427-432.

[10] Sanjay Chemawat, Howard Gobioff, and Shun-Tak Leung, "The Google File System," In 19th Symposium on Operating Systems Principles, pp. 29-43, Lake George, New York, 2003.

[11] Christian Cachin, Idit Keidar, and Alexander Shraer, "Trusting the Cloud," 2009ACM SIGACT News, Volume 40, Issue 2, pp. 81-86. 2009.

[12] Dmitry L. Petrov and Yury S. Tatarinov, "Data migration in the scalable storage cloud," 2009 International Conference on Ultra Modern Telecommunications & Workshops. pp. 1-4.

[13] David Patterson, Michael Armbrust, Armando Fox, etc, "Above the Clouds: A Berkeley View of Cloud Computing," Technical Report No.UCB/EECS-2009-28, UC Berkeley Reliable Adaptive Distributed Systems Laboratory. February 10, 2009.

[14] Giuseppe DeCandia, Deniz Hastorun, and Madan Jampani, etc, "Dynamo: Amazon's highly available key-value store," In SOSP, pp. 205-220, 2007.

[15] Avinash Lakshman and Prashant Malik, "Cassandra: A Structured Storage System on a P2P network," PODC `10 Proceeding of the 29th ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing. pp. 5-5.

[16] Xavier Defago, Peter Urban, Naohiro Hayashibara, and Takuya Katayama, "The Φ accrual failure detector," In RR IS-RR-2004-010, Japan Advanced Institute of Science and Technology, pp. 66-78, 2004.

 ISBN: 978-1-61208-106-9

An Active DBMS Style Activity Service for Cloud Environments

Marc Schaaf*, Arne Koschel*, Stella Gatziu Grivas†, Irina Astrova‡
*Department of Computer Science
University of Applied Sciences and Arts, Hannover, Germany
marc@marc-schaaf.de,akoschel@acm.org
†Institute for Information Systems
University of Applied Sciences Northwestern Switzerland, Olten, Switzerland
stella.gatziugrivas@fhnw.ch
‡Institute of Cybernetics
Tallinn University of Technology, Tallinn, Estonia
irina@cs.ioc.ee

***Abstract*—We propose an activity service as a component in cloud computing with the particular novelty that we base this service on the well-defined and proven semantics of Active Database Management Systems (Active DBMS). In addition, we utilize well-known principles of service oriented architectures. Furthermore we aim to provide an integration with a cloud service mediation component to automatically react to changes occurring in the cloud environment and in this way to implement agility and self management of cloud applications. As contribution of this paper we provide the high-level design of this activity service. This includes architecture, core interfaces and a semantically well-defined rule and execution model, based on extended Active DBMS semantics.**

***Keywords*-Active Database Management System (Active DBMS); Activity Service; Event Condition Action (ECA) Rules; Cloud Computing.**

I. INTRODUCTION

Cloud computing [1] is a 'trendy new kid on the block' as many recent activities in research and industry show. Companies and open source players almost constantly announce new features for their cloud platforms.

Event-based active mechanisms are an important feature for the cloud, be it just in form of messaging or in more elaborated event- or rule-driven behavior [2]. Although the necessity of supporting active behavior is clear, the open issue is the lack of a well-defined semantic.

The overcome of this drawback is the contribution of our work. We propose an activity service for cloud computing that adopts its semantic from the well-proven and clearly defined semantic of Active Database Management System (Active DBMS) style [3], [4] event-condition-action (ECA) rules and extend them for the cloud. Moreover, the design of the activity service is going to be based on proven principles and patterns from service oriented architectures (SOA, [5], [6]). Eventually, we will deliver an activity service with a precisely defined Active DBMS style ECA rule and execution model for the cloud.

The remainder of this article is organized as follows: The next Section will discuss related work. Afterward, Section III introduces Active DBMS style ECA rule processing. This is followed by an introduction of relevant cloud computing concepts. Section V joins both concepts to eventually provide an Active DBMS style ECA rule activity service for the cloud. We contribute the high-level design of this activity service, including architecture, core interfaces, and a semantically well-defined Active DBMS style rule and execution model, that is extended into the cloud.

II. RELATED WORK

Work, which is related to ours, occurs in different areas. Distributed event monitoring, which is an important part of our system, is an excellent instrument for (distributed) monitoring systems, see [7], [8] for overviews, and can contribute general monitoring principles to our work. However, these systems mainly concentrate on primitive event sources. Our work deals with event sources that are typically found in quite heterogeneous cloud computing environments. Event modeling aspects and semantics often lack precision [8] when compared to systems such as Active DBMS. Nevertheless, general work on the design of monitoring services for distributed systems is valuable for transfer into a cloud-based environment. Some event monitoring and propagation within the cloud in conjunction with complex event processing (CEP, [9]) is discussed in [10]. However, ECA rule processing with precisely defined semantics is not its focus.

The precise semantical foundation of our work is based on proven research work from the area of Active DBMS [3], [4]. In particular, we can utilize the Active DBMS manifesto [11]. This manifesto provides a proven ECA rule and execution model with a well-defined, clear semantic. Active DBMS style ECA rule processing will be discussed further in Section III and utilized in Section V.

One step beyond the work in Active DBMS go approaches concerning ECA rule processing in distributed environments. In [12], active functionality was extended into an ECA rule service for CORBA-based distributed, environments. Actually, this approach is one initial step in our direction.

 ISBN: 978-1-61208-106-9

A first step into ECA rule processing within cloud computing is done in [13]. At least some combination of event driven and service-oriented architecture for the cloud is discussed there. However, the work remains quite high-level and in particular focuses on policy-driven event processing for the cloud. It does not really address an activity service for the cloud at all, in particular not with well-defined Active DBMS style semantics.

Web services development standards such as the business process execution language WSBPEL [14], usually operate on a higher level than our approach. However, they are an excellent example for Web Services-based systems, that can generate events such as 'process' or 'activity' started/ended etc. In our approach, we have to deal with events across the heterogeneous cloud services and we must monitor and handle events generated by Web Services-based systems.

III. AN ACTIVE DBMS STYLE ACTIVITY SERVICE

The semantic foundation of our work is based on well-established earlier work from Active DBMS [3], [4], [11]. An Active DBMS is a standard 'passive' DBMS that has the capability to react to events based on event- condition-action (ECA) rules. The Active DBMS monitors the relevant events and notifies the component responsible for executing the corresponding rules (event signaling), which triggers these rules into execution. Rule execution incorporates condition evaluation as a first step and, if successful, action execution as the second step. A variety of execution models exist for the coupling of the transactions that raise events, evaluate conditions and execute actions.

An Active DBMS provides a rule definition language as a mean to specify event types, conditions, actions, and their assembly into ECA-rules. Execution constraints determine the coupling of events, condition evaluation and action execution within and across transactions. Binding information determines the granularity of the data items with which an event is associated. Information on event consumption determines how component events contribute to composite events and how event parameters enter into the computation of the composite event parameters. The rule base of an Active DBMS contains meta information on defined ECA-rules.

Our present work, in particular, follows the Active DBMS manifesto [11], which provides an established ECA rule and execution model with a well-defined semantic. Certainly however, this model requires extensions to take a cloud-based, distributed environment into account. However, the model already includes parameters such as event granularity information or event consumption policies, specifies different coupling modes etc. Such parameters can be summarized as *ECA semantic parameters*.

For traditional Active DBMS the mentioned functionality is usually closely tied to the DBMS. This is due to the usually monolithic system architecture of Active DBMS. Therefore it is quite hard to use their active functionality standalone in other contexts. For this reason, the active functionality was unbundled from Active DBMS to be usable as an activity service in other contexts [15]. It provides connectors for event detection, condition evaluation, action execution and an activity service exposes this as an overall functionality for active ECA rule processing. Now this unbundled active functionality is going to form a solid starting point for our present work as well.

IV. CLOUD COMPUTING CONCEPTS

Cloud computing has emerged as a technology becoming quite popular among companies and business. Computing resources like infrastructure, middleware or database functionality but also applications are provided over the Internet rapidly to users according to actual demands. The delivered resources are governable to ensure requirements like high availability, security, and quality. The key factor is that they are rapidly scalable up- and downwards, therefore the right amount of needed resources can be provided to the users.

Cloud Computing is a new paradigm, a new model based on known technologies like virtualization. What's new is the fast development and deployment of cloud applications. This is the contribution of the cloud computing to agility (fast response/reaction to new requirements, changes in the customer environment). The central element is the predictive management of the whole life cycle of a cloud environment, which is also the challenge. This subsumes all tasks like configuration, scale up/down and charge back.

We consider our work on the activity service as an important contribution towards the support of the predictive management of the cloud environment. For this, we propose a cloud broker or cloud mediator supporting functionality as reported in [16]. In [17], we discuss the extension of the functionality of a cloud broker, which has the intelligence to react to the changes of the business processes or their environment in order to change the cloud configuration (to scale up and down or to choose a new provider). The basis for the implementation of this functionality is the activity service we present in this paper, which could be, for example, used to monitor the utilization of the used services by evaluating corresponding business events. Based on the monitoring results the activity service could inform the cloud mediator that a service is at its capacity limits. In turn the cloud mediator could decide to switch to another service provider to increase the capacity.

V. AN ACTIVE DBMS STYLE ACTIVITY SERVICE FOR THE CLOUD

The aim of our work is to adapt the notion of an Active DBMS like activity service to the cloud. Therefore, we propose to place the required components for active mechanisms in the cloud and to thereby provide the mechanisms of

 ISBN: 978-1-61208-106-9

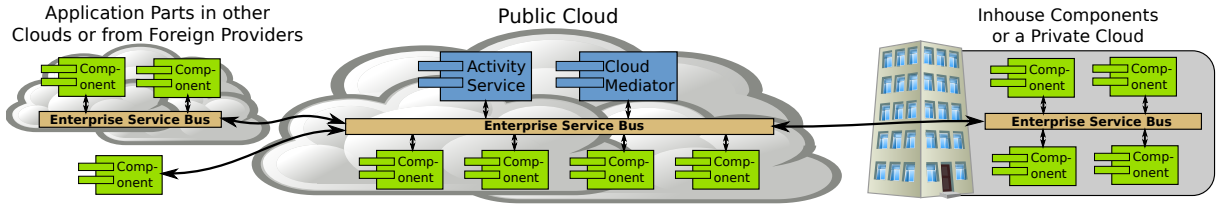


Figure 1. The High Level Architecture

an Active DBMS style *activity service* in this environment (Figure 1).

In order to achieve a high flexibility, the active mechanisms follow the unbundling approach mentioned in Section III. They are thus separated into different components. Each of the components provides one or more well-defined interfaces with clear semantics. Thereby the concrete implementation of the different components is interchangeable.

The communication between the components is realized based on the concept of a service oriented architecture (SOA). An Enterprise Service Bus provides means for the communication between the components in the cloud.

In our approach the event producers and consumers are not limited to the components in the cloud where the activity service is located. It can also gather information from other environments like from components in a private cloud of a company or from other clouds provided by other vendors.

Possible application areas for the activity service include the processing of vast amounts of events, which occur, for example, in logistics or finance applications. As mentioned the activity service can also be used for cloud monitoring purposes like for example for the automatic monitoring and scaling of a cloud application where the monitoring would be based on the evaluation of events from the different application parts.

A. The Components of the Activity Service

Figure 2 illustrates the different components of the activity service and their interactions. Their functionality is explained in the following subsections.

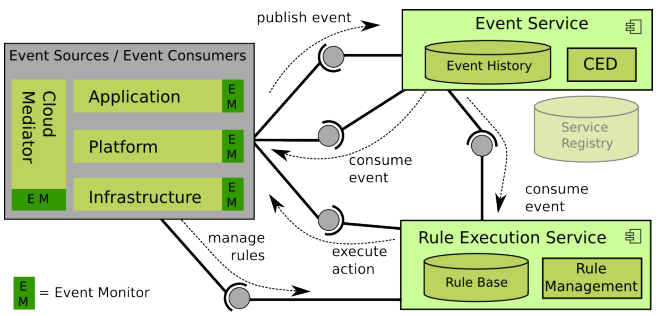


Figure 2. The Components of the Activity Service

1) The Event Service: The event service component implements all activities necessary for the cooperation between event producers and event consumers. It provides a service with the following interface, which can be used by event producers to send their events to the event service:

```
interface EventService{
  void sendEvent (Event)
}
```

For each incoming event, the event service determines if there are *event consumers* that are interested in this particular event and delivers the event to them. In addition, the incoming events are stored into an event history to support the monitoring of complex/composite events. A complex event detector (CED) evaluates the events and derives new complex events (see below), which are fed back into the processing mechanism. Consequently they are handled again as if they where incoming events.

To receive events from the event service an event consumer has to implement an appropriate *event handler service*, which needs to be published to the service registry. The event service discovers those services through the service registry. To inform the event service about the events a handler service is interested in, a filtering criteria has to be added to the WSDL description, which will be extracted by the event service.

Detection of Complex Events: Much work has been done in Active DBMS research regarding the detection of so called complex events ([8], [18]). Complex events are expressions of an event algebra, which are formulated over primitive or complex event types by means of algebraic operators. Say E1 and E2 are event instances. Complex events are then for example:

- disjunction (E1 $\vee$ E2), thus E1 or E2 occurred;
- conjunction (E1 $\wedge$ E2): E1 and E2 occurred, independent of their sequence;
- sequence (E1, E2): First came E1 then E2 occurred.

To detect complex events, basically, two techniques can be distinguished [18]:

- Backward discovery. In this technique upon arrival of a new primitive every event in the history of currently available events are checked, whether they together with the new event form a new complex event.

 ISBN: 978-1-61208-106-9

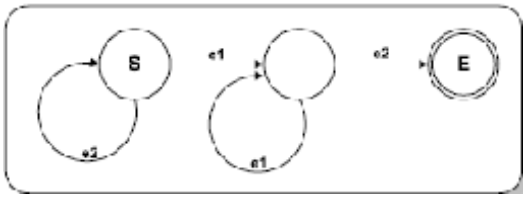


Figure 3. Complex event detection with a finite state automate

- Forward discovery: For forward discovery of complex events a sub-detector exists for every admissible complex event. Complex events are – without going back to the event history – detected in stages. The sub-detectors each have discovery status of 'their' complex event. The arrival of a primitive event leads to another step or a state change within individual sub-detectors. A complex event is detected, whenever a sub-detector reaches its final state.

In Active DBMS, a number of technologies for the discovery of complex events are used, such as finite state automates, Petri nets and event trees. As an illustrative example, Figure 3 shows a finite state automate. It detects a complex event: sequence E = (e1, e2) with start state S and final state E.

We currently aim to integrate backward discovery into the Event Service. A decision on the concrete technology for a prototypic implementation is yet to be made. However, the complex event detection process is hidden from the service consumers and can thus easily be changed to a forward discovery based approach if required.

2) The Rule Execution Service: The rule execution service receives events from the event service to evaluate them against sophisticated ECA rules. Therefore it acts as an event consumer of the event service by registering an event handler service. The rules result in the execution of action handlers. Such an *action handler* needs to be implemented by each of the components that are intended to be called from within rules. The rule can also provide the action handler with parameters, which can be derived from the rule execution.

The rules that are evaluated are stored in a rule base, which can be managed by a special *rule management service*. Using the rule base, the rules are implemented by the rule execution service.

3) Event Monitors: Normally, not all components are build for active notification by the event service. For other components a *monitor capsule* mechanism is possible. Therefore a small application that acts as a capsule around the actual event source can be realized that obtains the event from the source and transfers it to the event service. In addition, the conversion between different event types can be realized by the capsule.

To further illustrate the event monitoring, a concrete example for a particular kind of event source will now be given. In particular, we utilize earlier work from us [19] to take a look at the monitoring of an 'active' event source, here the Oracle 10 relational DBMS (Figure 4).

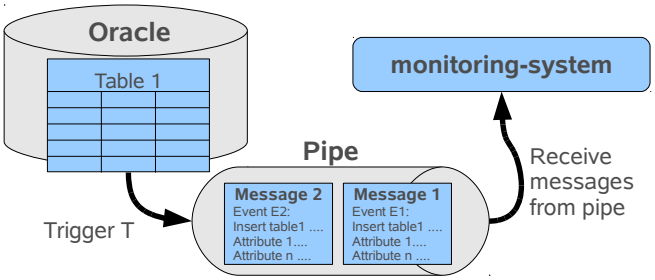


Figure 4. Monitoring using Triggers and Pipes

The system allows for communication between Oracle sessions by means of so called Oracle Pipes. A pipe is a data structure into which messages may be placed and from where they may be retrieved in FIFO order. If the pipe is empty, a recipient is blocked until a new message is available.

Note that a message in a pipe becomes immediately visible independent of the status of the transaction that placed it there. Even if the transaction that produced the event is aborted, the event may is already passed onto further processing.

A pipe is a communication structure, which naturally fits into cloud computing. In particular larger cloud providers, e.g., Amazon's EC2/S3 or Microsoft's Azure provide cloud messaging queues. Our event monitor thus would read the events from Oracle pipes and place them into Cloud messaging queues, which are connected to the ESB from our activity service, for further processing.

Now suppose that an event type is defined for the source, say insertion of a tuple into some relation. Our monitor capsule internally declares a corresponding Oracle trigger which, when fired, places a message with all relevant information into the pipe. The capsule thread then acts as the recipient and hands the event over to the activity service.

The call-back mechanism has the advantage of event detection without delay and incurs negligible overhead because no query processing is needed in contrast to several other mechanisms, which are discussed in [19]. A significant drawback though, is the lack of standardization for the call-back mechanism and, hence, its limited availability.

Looking at the Active DBMS style ECA semantic parameters (see below), the natural coupling mode for this type of event sources is 'immediate decoupled' due to the independence of pipes from transactions.

All other modes require some additional effort. For example, to support the 'immediate coupled' mode, a second pipe must be installed, which receives the event signaling completion of the sub-transaction. A trigger is defined as the recipient and takes the appropriate action for the main

 ISBN: 978-1-61208-106-9

transaction. In order to support the 'deferred decoupled' mode, a wrapper thread must observe a second structure in which the DBMS must make note of the completion of the transaction. Unfortunately, this solution entails some overhead due to the need for polling.

B. The Rule and Execution Model

Since our work follows the Active DBMS style rule and execution model from the Active DBMS manifesto we plan to investigate the semantic parameters for their suitability for cloud environments. It is expected that some modifications need to be made. For example, we can only provide certain transaction coupling modes, since we can't assume all cloud event sources to be able, to participate in 2-phase-commit transactions. Similarly, some event sources might only be able to send events as a whole rather than individually, so the event (sending) granularity might be event source dependent.

Beside evaluating, which ECA semantic parameters still fit for the cloud, we are going to investigate extensions of those parameters, to fit even better into cloud computing. Additional parameters include for example: Options to execute ECA rules across different rule processors within the cloud, configurable reactions to deferred events due to network issues, options to deal with not responding event or data sources and several more like cost parameters, which allow to choose for example an especially cheap activity service or an especially good one, etc.

VI. CONCLUSION

With this work, we aim to create a complete and versatile concept, which reaches from the event monitoring over the preprocessing to the evaluation of sophisticated rules and as a reaction the execution of actions on cloud components. Furthermore our whole concept is based on the well-defined and proven semantics of Active DBMS style ECA rule processing, which other event processing approaches lack (cf. Section II). To provide a maximum of flexibility we utilize the well-known principles of service oriented architectures and provide the different functionalities in different interchangeable components with well-defined interfaces and clear semantics and aim to support heterogeneous event sources by the concept of monitoring capsules. Furthermore we intent to provide an integration of the activity service with a cloud mediator so that the mediator can utilize event information to react immediately to changes in an applications environment.

Our next steps will be the detailed specification of the components, their interactions and interfaces and especially the adoption of the proven, semantically rich Active DBMS style rule and execution model to the new world of cloud computing. Moreover we will work on the specification of an appropriate rule and event definition language. Finally we will provide implementations of the different components of the activity service and evaluate their potential in real world application scenarios.

ACKNOWLEDGEMENT

Irina Astrova's work was supported by the Estonian Centre of Excellence in Computer Science (EXCS) funded mainly by the European Regional Development Fund (ERDF).

REFERENCES

[1] M. Armbrust *et al.*, "Above the Clouds: A Berkeley View of Cloud Computing," EECS Department, University of California, Berkeley, Tech. Rep., 2009.

[2] S. Rozsnayi *et al.*, "Event Cloud - Searching for Correlated Business Events," *9th IEEE International Conference on E-Commerce Technology*, 2007.

[3] J. Widom and S. Ceri, Eds., *Active Database Systems: Triggers and Rules for Advanced Database Processing*. San Francisco, CA, U.S.A: Morgan Kaufmann Publishers, 1996.

[4] N. W. Paton, Ed., *Active Rules for Databases*. New York: Springer, 1999.

[5] D. Krafzig, K. Banke, and D. Slama, *Enterprise SOA: Service-Oriented Architecture Best Practices*. Upper Saddle River, NJ: Prentice Hall, 2004.

[6] T. Erl, *SOA Design Patterns*. USA: Prentice Hall, 2009.

[7] B. Schroeder, "On-Line Monitoring: A Tutorial," *IEEE Computer*, vol. 28, no. 6, pp. 72–80, Jun. 1995.

[8] S. Schwiderski, "Monitoring the Behaviour of Distributed Systems," Ph.D. dissertation, Selwyn College, University of Cambridge, United Kingdom, 1996.

[9] D. C. Luckham, *The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2001.

[10] G. Wishnie and H. Saiedian, "A complex event routing infrastructure for distributed systems," vol. 2. Los Alamitos, CA, USA: IEEE Computer Society, 2009, pp. 92–95.

[11] The ACT-NET Consortium, "The Active Database Management System Manifesto: A Rulebase of ADBMS Features," *ACM SIGMOD Rec.*, vol. 25, no. 3, pp. 414–471, Sep. 1996.

[12] A. Koschel and P. C. Lockemann, "Distributed events in active database systems: letting the genie out of the bottle," *Data Knowl. Eng.*, vol. 25, no. 1-2, pp. 11–28, 1998.

[13] P. Goyal and R. Mikkilineni, "Policy-based event-driven services-oriented architecture for cloud services operation & management." Los Alamitos, CA, USA: IEEE Computer Society, 2009, pp. 135–138.

[14] WSBPEL TC, "Web Services Business Process Execution Language Version 2.0," OASIS, OASIS Standard, 2007.

 ISBN: 978-1-61208-106-9

[15] S. Gatziu, A. Koschel *et al.*, "Unbundling active functionality," *ACM SIGMOD Rec.*, vol. 27, no. 1, pp. 35–40, 1998.

[16] L. Leung, "Cloud computing brokers: A resource guide," 2010, url: http://www.datacenterknowledge.com/archives/2010/01/22/cloud-computing-brokers-a-resource-guide/ Visited: 10.06.2010.

[17] S. Gatziu, T. U. Kumar, and W. Holger, "Cloud Broker: Bringing Intelligence into the Cloud An Event-Based Approach," in *Proc. of the 3rd IEEE Intl. Conf. on Cloud Computing*, Miami, Florida, July 2010.

[18] K. Dittrich and S. Gatziu, *Aktive Datenbanksysteme, Konzepte und Mechanismen*. Int. Thomson Publishing GmbH, Bonn, Albany, Attkirchen, 1996.

[19] A. Koschel and I. Astrova, "Event monitoring web services for heterogeneous information systems," *Proc. World Academy Of Science, Engineering And Technology*, vol. 43, pp. 50–52, 2008.

 ISBN: 978-1-61208-106-9

Introducing a Dynamic Federation Model for RESTful Cloud Storage

Yang Xiang
Rechenzentrum Garching
Max-Planck-Society
Garching, Germany
yang.xiang@rzg.mpg.de

Sebastian Rieger
GWDG
Max-Planck-Society
Göttingen, Germany
sebastian.rieger@gwdg.de

Harald Richter
Department of Informatics
Clausthal University of Technology
Clausthal-Zellerfeld, Germany
hri@tu-clausthal.de

***Abstract*—This paper presents a solution for RESTful cloud storage in a dynamic identity federation. With dynamic federations, Cloud Service Providers are able to find Identity Providers autonomously in the cloud in order to make services flexible, scalable and interoperable. By combining a Representational State Transfer architecture with SAML-based identity federation, a distributed and decentralized cloud storage is provided which allows users to access files via the Internet seamlessly and transparently.**

***Keywords*-Dynamic Identity Federation; REST; Cloud Computing; Storage; SAML**

I. INTRODUCTION

Cloud computing is a modern way to provide IT services as a kind of commodity to customers via the Internet. In storage clouds, which are a specialization of cloud computing, files can be kept at different sites, and the user is able to mount his virtual storage from any computer connected to the Internet independently of which site actually hosts the data. This is called transparent and seamless file access and it is usually accomplished by offering a web interface to the user.

This paper introduces a new solution for authentication and authorization (AA) for storage clouds which employ dynamic identity federation. Its focus lies on contemporary storage clouds such as Amazon S3 [1] and Google Storage [2]. Both clouds are decentralized, web-based and use the Representational State Transfer architecture (REST) [3] as a communication framework between the server(s) of the storage cloud and the users. Though having commonalities, S3 and Google Storage are incompatible with each other. In the following, a model for dynamic federation is described that simultaneously supports multiple cloud service providers (CSPs) by augmenting "RESTful" storage clouds. Here RESTful means that the clouds have to be compatible with protocol definitions and constraints according to REST. However, different storage cloud providers normally do not federate their services, thus user files stored on a given provider will not be accessible from another provider. The model which we present here has the advantage that a user will be presented with a single means of access which spans CSPs. Furthermore the user has more flexibility and can change CSPs completely or switch between CSPs temporarily at will. However, more flexibility for the user also requires a faster establishment of trust between user and CSP. Establishing, measuring and predicting trust in an automatic manner is one of the targets of our model. Please note that this model is not intended to act as an identity management system, and that it is not limited to storage clouds only.

Since decentralized AA, as used in identity federations, offers a unified means of authentication and authorization across different storage cloud providers, both S3 and Google Storage can be accessed in a uniform manner. Here, it is necessary to focus on web browser clients and on RESTful file access together with virtual file systems as described by the Storage Cloud Initiative of SNIA (Storage Networking Industry Association) [4]. Our model follows this industry standard and implements a unified AA for cloud-based storage solutions.

A. State-of-the-Art

In recent years, RESTful web services have gained popularity and may be a potential alternative to SOAP solutions [5][6]. Compared to SOAP, REST directly uses HTTP methods to transfer data between client and server without much protocol overhead. REST is less complex and thus less resource-intensive than SOAP which is the reason why it has gained interest over SOAP. REST data structures can be encoded in HTML or XML, and RESTful web services are easily understandable and human-readable since they are reduced to a minimum and in plain text. S3, for example, uses HTTP PUT, GET and DELETE methods in a RESTful style to read, write and manage files via so-called buckets [1]. Access control, i.e., file access authorization is accomplished by extra protocol data called authorization header that contains all user credentials. This header and the REST-based file access structure of S3 is shown in Figure 1 as an example. Other cloud storage providers such as Ubuntu One [7] use the same technique or even extend the REST functionality by using WebDAV [8]. Regarding the prevalence of RESTful applications in contemporary storage clouds, REST could become a basis also for future clouds which is why our AA system uses this technology. However, the RESTful approach in S3 has two major drawbacks: First, the user needs a special client or middleware that is able to create and send the proprietary authorization header to the CSP. As a result,

 ISBN: 978-1-61208-106-9

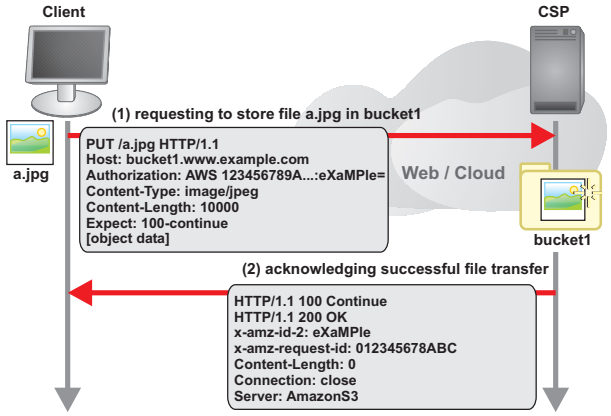


Figure 1. Basic access structure of Amazon S3

the interoperability between different CSPs is low. Second, the user needs multiple credentials, one for every CSP he uses. While the interoperability problem is being addressed by SNIA in their CDMI standard [4], the multiple credential problem still limits the simultaneous usage of different storage clouds. Furthermore, if all users of a storage cloud are not under the umbrella of the same scientific or commercial organization, difficulties to authenticate and authorize them arise and the following two problems have to be solved: How to authenticate an individual user accessing the service anonymously from the Internet? How to manage AA of thousands of users that belong to different organizations which are all different cloud customers? As a solution to these problems, an extension to existing AA infrastructures is proposed that is based on identity federation.

In an identity federation, a CSP does not have to care about the user's identity by itself. Instead, it delegates this task to a so-called identity provider (IdP) which is responsible for the user, and who acts as a user proxy. The idea is to augment the identity management system of a CSP by the AA concept of an identity federation. For this purpose, we suggest a "trust estimation system" (TES) as a key stone of our model to quickly establish and estimate the quality of the trust relationship between CSP and user that computes numerical quantities for the representation of trust and reputation. Our TES uses SAML [9] for communication and can be easily incorporated into a Shibboleth [10] IdP or SP. SAML was chosen because it is widely supported by major service providers as their de-facto AA standard.

The remainder of the paper is organized as follows: in Section 2, related work is presented. In Section 3, the requirements for identity federations that hold in storage clouds are defined. Section 4 describes the set-up of the proposed dynamic identity itself. Section 5 presents the TES and its implementation. In Section 6, conclusion and future work are given.

II. Related Work

There are efforts to combine the advantages of REST and SAML. A US patent [11], for instance, describes a method to securely invoke a REST API-call by means of a SAML security token. With this method, a client obtains a security token from an authentication server beforehand. When a user sends a request to an application server to invoke a REST API-call, authentication is performed by means of an HTTP-digest. After successful authentication, the client computes a token digest by using the security token it has received from the authentication server, together with a NONCE (number used once) and a time stamp from the application server. Finally, the client sends this token digest back to the application server. This method is considered good but not appropriate for the environment of CSPs.

There are a few open source projects that are focusing to provide cloud-based virtual file systems, for example iRods and GridFS from mongoDB [12], beside the large commercially available clouds Amazon S3 and Google Storage. Furthermore, SAML-based solutions for identity federation such as Shibboleth have been used already in several research projects in order to provide federated AA to users of iRods [13]. However, such AA is based on statically-federated Shibboleth-implementations only, and the cloud storage is also not RESTful in these projects.

III. Requirements for Identity Federations in Storage Clouds

Large scientific and commercial communities typically have a spatially distributed structure. The information and communication equipment may be disjoint and in part even incompatible between organizational branches. Different hardware, operating systems and middleware may be in use, as well as different storage techniques, either on the record level or the file system level. Furthermore, different qualities in trust and reputation may exist for users and user communities, as a result of how they have behaved in the past. Regarding this situation, there are several requirements for AA in storage clouds:

- AA should serve for multiple storage clouds simultaneously, and the designated AA mechanism should support multiple users in each identity federation.
- There should be no central AA instance because this would be not scalable and a single-point of failure as well. A distributed AAI is needed instead.
- There shall be a mechanism for adding and removing users dynamically for short-term projects which is similar to virtual organizations (VOs) known from grid computing.
- Each home institute or subsidiary may use a different AA system. For all institutes that are not willing to use SAML, a backdoor solution should be available by means of an AA gateway that translates local AA tokens into SAML.
- AA should also work together with CSPs that do not support entering the home organization of a user in the CSP's web form. As a consequence, direct file access without a web form is needed. This means that the global verification of locally known user names and passwords must work under all circumstances. Therefore, a mechanism is sought to automatically detect the user's

 ISBN: 978-1-61208-106-9

home organization, instead of requesting him to select manually his home organization in a web form as is common in current practice.
- A trust estimation service (TES) is required which can ensure that identity assertions truly come from an IdP as a user proxy rather than from an intruder. The TES must be resistant against several types of attacks.
- Users must be able to apply the same credentials they gained from their home institute or subsidiary for accessing different CSPs (i.e., single sign-on).
- User credentials must not be transmitted beyond an organizational border in non-encrypted form in order to protect privacy. Furthermore, to make the communication secure, data transfer between identity providers (IdPs) and cloud service providers (CSPs) must be encrypted to prevent user name and password phishing.
- Access to user files should be based on existing de facto standards of storage clouds. Furthermore, AA should be compatible with existing file system and file-backup tools.

In our opinion, these requirements lead to a need for a dynamic federation model, where values for trust and reputation are calculated quickly and automatically for each user and IdP. More information about the means of calculating the trust and reputation values can be found in Xiang et al. [14].

IV. Identity Federations

The following two subsections illustrate the advantages of dynamic federations, as presented in this paper, over existing static federation solutions.

A. Static Identity Federation

There are different types of static identity federations with respect to the underlying software technologies. For example, OpenID and Windows Card Space belong to the user-centric category, while Shibboleth is an institution-centric system. However, both categories exhibit the same problems which are listed as follows:

1) There is substantial manual operator effort to maintain existing identity federations which lies in the order of $O(n^2)$, resulting from the fact that trust relationships between every pair of CSPs and user must be defined.
2) Trust relationships are expressed by static values only. This limits the cloud's scalability since increasing the number of users becomes very time-consuming.
3) It is difficult or impossible to dynamically connect independent federations to form a confederation because an entity from one federation does not know and hence does not trust entities from the other federations.
4) If a major customer of a CSP wants to resell the service he obtains from his CSP to his own sub-customers, then the CSP has to add all sub-customers to its trusted IdP list. No hierarchic linking is possible.
5) The costs of adding customers into an identity federation increases substantially if customers join and leave the cloud at a high rate. This is a hindrance for clouds to sell storage as a commodity.

B. Dynamic Identity Federations

In a dynamic identity federation, a CSP does not need to know an IdP beforehand. A trust relationship is created on demand, and a corresponding trust value will be determined on-the-fly. This is beneficial to those CSPs who want to provide identity as a service (IDaaS) as a new business model in cloud computing. CSPs which are specialized on IDaaS act as an identity provider for other CSPs that want to sell more elaborated services such as infrastructure as a service (IaaS), platform as a service (PaaS), or software as a service (SaaS). This means, CSPs for IDaaS bridge the gap between end-customers and other CSPs that are specialized in IaaS, PaaS or SaaS. As a result, a new business model may be established since CSPs for IaaS, PaaS and SaaS can simply delegate AA to an IDaaS provider. Hence, the 1:n relationship of an existing static federation is reduced to a 1:1 relationship in a dynamic federation as shown in Figure 2 . In Figure 3, the

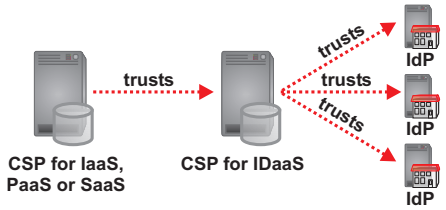


Figure 2. CSP for IDaaS acts as an identity store for other CSPs

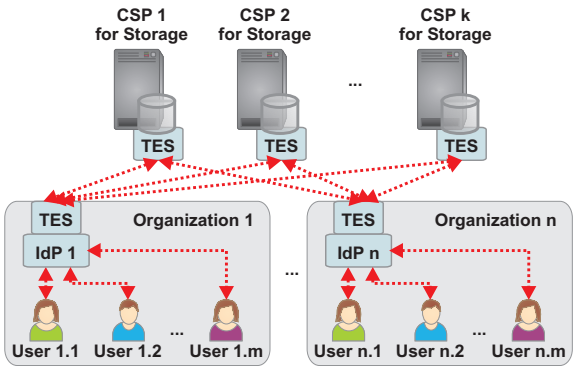


Figure 3. AA for storage clouds serving multiple identity federations

dynamic identity federation and the TES for storage clouds are illustrated. It supports AA for multiple storage clouds that in turn serve multiple organizations or institutions which form a dynamic federation. AA inside an organization or institution is accomplished via local IdPs which are responsible for subsets of users. The TES performs the communication between IdPs and CSPs and computes trust values. in Figure 4, an arbitrary organization i and a random user j inside of i are chosen as an example scenario in order to explain the function of the TES. In this figure, the individual protocol steps between user, IdP and CSP are as follows:

1) User $i.j$ shall be a customer of CSP k which has electronically signed a contract with IdP i that all users from organization i are allowed to access the storage

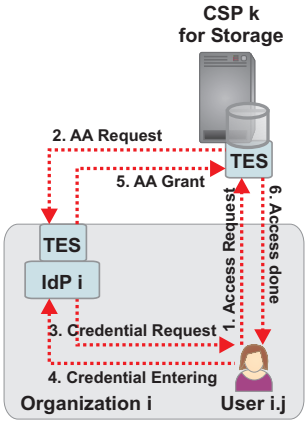


Figure 4. AA protocol inside an identity federation with TES

service of CSP k. Now, user $i.j$ is requesting access to a file at CSP k.

2) Then, CSP k determines that IdP i is responsible for user $i.j$ by means of a dynamic discovery service (DDS) - that is part of our TES - and asks IdP i for authentication, instead of authenticating $i.j$ by itself.
3) IdP i requests user $i.j$ to enter his user name and password.
4) User $i.j$ supplies his username and password to IdP i.
5) After the IdP i has successfully authenticated user $i.j$ it will send a positive response back to CSP k. CSP k may then request more attributes of user $i.j$ from IdP i for subsequent authorization.
6) Finally, the file access is executed by CSP k and the operation is acknowledged positively or negatively to user $i.j$.

After the first successful AA procedure, user $i.j$ can access his files without reentering his user name and password as long as the HTTP session with the IdP persists. With identity federation, CSPs and IdPs trust all entities in that federation more or less, varying on their trust value. The trust value the user's IdP i has with respect to the user's CSP k is estimated by the TES.

V. Implementation of Dynamic Identity Federations with TES for RESTful Cloud Storage

In the next two subsections, the structure of the TES will be presented.

A. Extending Shibboleth with a TES

For our TES prototype, Shibboleth was chosen as a basis because it is widespread and functional. Shibboleth is a SAML-based framework for static identity federations and consists of two parts, an IdP which is written in Java, and a SP which is implemented in C++. It can be used as an authentication module for Apache web servers. In Shibboleth, the trust relationship between entities is established with static meta data containing one or more X.509-certificates. To make Shibboleth dynamic by exchanging meta data on demand and in real time, Shibboleth is extended by a TES as shown in Figure 5. The data flow of the new functionality is as follows:

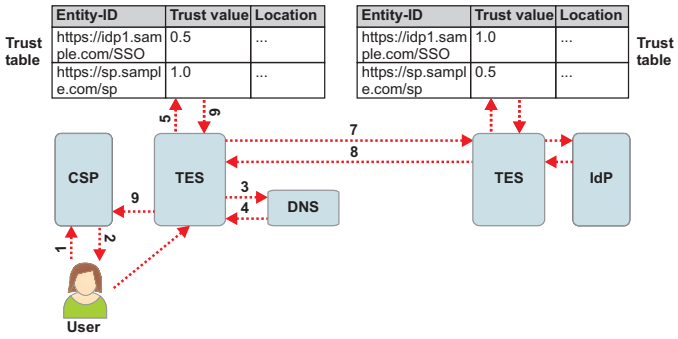


Figure 5. TES as a Shibboleth extension

1) When an end user accesses a CSP, he will be redirected to a dynamic discovery service (DDS) which is a part of our TES. He can then input his e-mail address (steps 1 and 2 in Figure 5).
2) After step 1, DDS/TES sends a query to the domain name system of the Internet to obtain the Entity-ID of the user's home IdP (steps 3 and 4 in Figure 5).
3) The obtained Entity-ID of the user's home IdP is used as a search criterion in a table of the local TES, and the local TES looks-up the corresponding end-location of the home IdP (steps 5 and 6). Every end-location is an Internet URL. For each IdP, which has already been queried, an entry will exist. Beside the IdP's end-location, each entry contains a trust estimation value for the IdP.
4) If the IdP can not be found in the trust table of the local TES, or if its trust value is lower than a configurable minimum, then the IdP is treated as not trustworthy, and the local TES issues an error message to the end user. If the IdP is found in the trust table, and if its trust value is above the minimum level, then the IdP is treated as trustworthy. Subsequently, the local TES sends a request to the IdP's TES in order to retrieve the IdP's meta data (steps 7 and 8).
5) Finally, the local TES forwards the IdP's meta data to the CSP (step 9).

The content of the trust table is propagated among the TESs by using a self-developed protocol [14], which is similar to OSPF. In doing this, the IdP's meta data is retrieved and updated dynamically by TES.

B. Using DNS NAPTR Record for Discovery Service

As described in the previous section, the Internet DNS is used to resolve the Entity-IDs of IdPs which are URLs. Since DNS is a distributed system, it matches the decentralized nature of the dynamic identity federations proposed here. In decentralized federations, each institution or organization maintains the Entity-IDs of its IdPs in local name server(s). That information is disseminated over the Internet to other entities. When a user requests a CSP service, then he enters his

 ISBN: 978-1-61208-106-9

email address via a dedicated user client or a web browser, and the DDS asks its local name server to resolve the Entity-ID of the user's home IdP. Although there is an existing approach for this described procedure [15] using DNS SRV records according to RFC 2782 [16], we chose another solution because DNS SRV records are limited in their capability to map a service onto a fully qualified URL. For example, DNS SRV records cannot contain paths like "http://idp.aai.mpg.de/idp/". Since most Entity-IDs have URLs with paths included, we opted for DNS NAPTR records instead, according to RFC 3403 and RFC 3404 [17]. Also OASIS [18] recommends NAPTR records for resolving meta data via DNS.

A DNS query by means of NAPTR records is employed to map pairs of URNs, URLs onto DNS domain names. The DNS query in turn returns a record that has the following data elements: a) Order, b) Preference, c) Flags, d) Services, e) Regexp and f) Replacement. The elements of this list have the subsequent meaning:

- Order field: A 16-bit unsigned integer specifying the order in which a set of NAPTR records must be processed.
- Preference field: A 16-bit unsigned integer specifying the order in which NAPTR records with equal order fields should be processed. Records with a lower value in the preference field should be processed before records with a higher value.
- Flag field: A <character-string> containing control bits for the rewriting and interpretation of the subsequent fields following the flag field.
- Service field: A <character-string> that specifies the parameters for this service, including the delegation path. The semantics of this field are service-specific.
- Regexp field: A <character-string> that contains a regular expression which substitutes the original input string obtained from the client to construct the proper domain name for address resolving.
- Replacement field: Contains the next domain name to be queried. Depends on the flags field.

An example of the NAPTR record is as follows:

```
$ORIGIN example.com.
IN NAPTR 100 10 "u" "aai+idp"
"!^.*$!http://aai.mpg.de/idp/!" .
```

This record has an order value of 100 and a preference of 10. The flag "u" is a terminal symbol and indicates that the output of the regular expression is a URI. The next field "aai+idp" indicates that this is an AA service and that the record deals with an IdP instead of a SP. The replacement field means that a domain name such as "mpg.de" has to be replaced by the full URL "http://aai.mpg.de/idp/".

C. Implementation of a RESTful Storage Client

The TES is designed as an extension to Shibboleth which requires a fully-featured web browser and interaction with the end user. In the sign-on process, the Shibboleth SP uses a so-called SAML HTTP POST profile [19] that needs support from JavaScript on the client side to automatically return the web form back to the IdP in order to get the user authentication data. The same JavaScript functionality is used to transmit the SAML assertion with one or more authentication statements back to the SP [20]. This works fine with all current web browsers that have JavaScript enabled. However, for some RESTful API-calls, manual user interaction via a browser are not possible. One solution to this problem is the enhanced client and proxy (ECP) profile defined in SAML [19], but ECP is currently offered only as an experimental extension to Shibboleth, and it is limited to SOAP. Another solution for transmitting the assertion back is an immediate URL-encoding in a HTTP redirect request, but this is not supported by Shibboleth [21]. Therefore, another method that does not need JavaScript was chosen. Here, the HTTP-Response with the SAML assertion is interpreted directly by the REST-client without JavaScript functionality. The client extracts the content of the web form using XPATH and forwards the assertion to the SP. The entire data flow between client, IdP and the CSP is depicted in Figure 6.

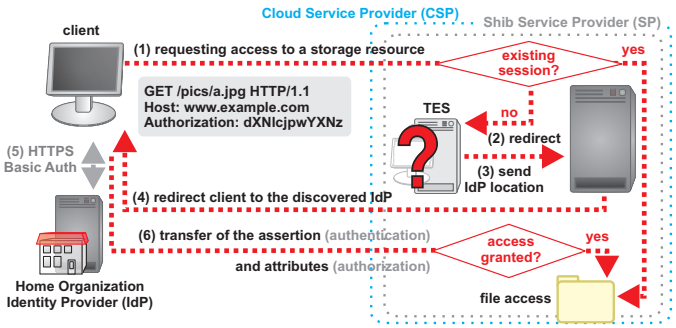


Figure 6. Data flow of federated access to cloud based storage solutions

As in Amazon S3, a base64-encoded authorization header is included in the HTTP request from the client to the CSP. The whole data flow comprises 6 steps which are described below:

1) In step 1, the client transmits an access request for file "a.jpg", for example, via a RESTful HTTP message. Its authorization header is compliant with the basic HTTP authentication process described in [22] and therefore uses HTTPS.
2) On the CSP side, first a check to determine if a session exists is performed. If none exists then the HTTP request is redirected to the TES. If a session already exists file access is granted.
3) If there is no session then the user name contained in the HTTP header (e.g., user@institute-a.mpg.de) is checked by the TES to discover the user's home organization and his home IdP. After having found the user's IdP, the TES sends the IdP's location to the CSP.
4) Subsequently, the client is redirected to the discovered IdP.
5) If the client was not previously authenticated at the IdP (no existing session at the IdP), the authorization header is used to authenticate the client.

 ISBN: 978-1-61208-106-9

6) After a successful authentication, the IdP transmits a response which contains the SAML assertion as a hidden HTML input field (SAML HTTP POST profile). Since our RESTful storage client may work independently of a web browser, the value of the hidden input field is extracted by using XPATH and forwarded to the CSP with a POST request. If the authentication was successful, the client is redirected again to the resource "a.jpg" which was requested by the end user, and the data flow cycle is completed.

The delegation of AA to the IdP has an important advantage: if a client accesses multiple cloud storage providers, within the same session, the client does not need to authenticate again. Hence, a single sign-on solution (SSO) is achieved. The same holds for private clouds, as well as for services beside storage, thus allowing for a flexible and comfortable exploitation of the cloud paradigm.

VI. Conclusion and Future Work

This paper describes the usage of dynamic identity federations together with a trust estimation system as an extension to Shibboleth. Furthermore, the integration of dynamic federation into a RESTful cloud storage environment is presented. A dynamic federation model allows for the discovery of the users' home IdPs by means of DNS NAPTR queries. In a distributed storage cloud, files are kept at different sites, and it is necessary to enhance the user client for proper handling SAML assertions. In our model, these assertions are contained in the HTTP-response and do not require web browsers which have JavaScript enabled. Furthermore, the model can be used in private clouds and we are currently evaluating a private storage cloud based on Eucalyptus Walrus [23] which is Amazon S3-compatible. Our next goal is to modify the existing client software and tools of Walrus to support federated authentication. Finally, because our proposed solution is not yet able to provide virtual file systems for common operating systems, future work is needed to adapt WebDAV clients to virtualize the access to dynamically federated cloud storage. Existing WebDAV clients support redirections without problems, but the extraction of assertions and the handling of the subsequent HTTP POST has still to be implemented. Last but not least, authorization issues beyond the available access control scheme of Shibboleth are not addressed yet. To implement fine-grained access control lists on the file level, e.g., according to the user attributes, it is necessary to integrate dynamically federated storage clouds more closely with the underlying operating systems.

References

[1] "Amazon Simple Storage Service (Amazon S3)." [Online]. Available: http://aws.amazon.com/de/s3/ [2010.06.05]

[2] "Google Storage for Developers - Developer's Guide." [Online]. Available: http://code.google.com/intl/en/apis/storage/docs/developer-guide.html [2010.06.16]

[3] R. T. Fielding, "Architectural styles and the design of network-based software architectures," Ph.D. dissertation, University of California, Irvine, 2000.

[4] "Cloud data management interface," SNIA Web Site, December 2010. [Online]. Available: http://cdmi.sniacloud. com/ [2010.06.16]

[5] F. AlShahwan and K. Moessner, "Providing SOAP Web Services and RESTful Web Services from Mobile Hosts," in *2010 Fifth International Conference on Internet and Web Applications and Services*. IEEE, 2010, pp. 174–179.

[6] C. Pautasso, "REST vs. SOAP: Making the Right Architectural Decision," in *SOA Symposium*, 2008, pp. 2009–01.

[7] "Ubuntu one." [Online]. Available: https://one.ubuntu.com/ [2010.06.22]

[8] L. Dusseault, "RFC 4918: HTTP Extensions for Web Distributed Authoring and Versioning (WebDAV)," RFC, IETF, June 2007., Tech. Rep.

[9] S. Cantor, J. Kemp, R. Philpott, and E. Maler, "Assertions and protocols for the oasis security assertion markup language (saml) v2. 0," 2005.

[10] "Shibboleth System." [Online]. Available: http://shibboleth. internet2. edu/ [2010.06.16]

[11] R. Lai and K. Chan, "METHOD AND APPARATUS FOR SECURELY INVOKING A REST API," Patent, Mar. 12, 2008, uS Patent App. 12/046,579.

[12] "Mongodb gridfs specification." [Online]. Available: http://www.mongodb.org/display/DOCS/GridFS+Specification [2010.06.15]

[13] "ASPiS: Architecture for a Shibboleth-Protected iRODS System." [Online]. Available: http://mykcl.com/iss/cerch/ projects/completed/aspis.html [2010.06.18]

[14] Y. Xiang, J. Kennedy, H. Richter, and M. Egger, "Network and Trust Model for Dynamic Federation," The Fourth International Conference on Advanced Engineering Computing and Applications in Sciences. IARIA, 2010.

[15] S. Rieger and T. Hindermann, "Dezentrales Identity Management für Web- und Desktop-Anwendungen," in *Proc. 1. DFN-Forum Kommunikationstechnologien, Kaiserslautern 2008. Gesellschaft für Informatik, Bonn, 2008; S. 107-116*.

[16] A. Gulbrandsen, P. Vixie, and L. Esibov, "RFC2782: A DNS RR for specifying the location of services (DNS SRV)," *RFC Editor United States*, 2000.

[17] M. Mealling, "RFC3403: Dynamic Delegation Discovery System (DDDS) Part Three: The Domain Name System (DNS) Database," *RFC Editor United States*, 2002.

[18] S. Cantor, I. Moreh, S. Philpott, and E. Maler, "Metadata for the OASIS Security Assertion Markup Language (SAML) V2. 0," 2005.

[19] S. Cantor, J. Hughes, J. Hodges, F. Hirsh, P. Mishra, R. Philpott, and E. Maler, "Profiles for the oasis security assertion markup language (saml) v2. 0," 2005.

[20] "SWITCH AAI Expert Demo." [Online]. Available: http://www.switch.ch/aai/demo/2/expert.html [2010.06.22]

[21] "Shibboleth Native SP Assertion Consumer Service." [Online]. Available: https://spaces.internet2.edu/ display/SHIB2/NativeSPAssertion ConsumerService [2010.06.22]

[22] J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen, and L. Stewart, "RFC2617: HTTP authentication: basic and digest access authentication," *Internet RFCs*, 1999.

[23] "Walrus Storage Service." [Online]. Available: http://open. eucalyptus.com/wiki/EucalyptusStorage_v1.4 [2010.06.22]

 ISBN: 978-1-61208-106-9

Open Architecture for Developing Multitenant Software-as-a-Service Applications

Javier Espadas, David Concha

Tecnológico de Monterrey, Campus Monterrey
Monterrey, México
{mijail.espadas,david.concha}@itesm.mx

David Romero, Arturo Molina

Tecnológico de Monterrey, Campus Ciudad de México
City, México
david.romero.diaz@gmail.com, armolina@itesm.mx

Abstract. **As cloud computing infrastructures are growing, in terms of usage, its requirements about software design, management and deployment are increasing as well. Software-as-a-Service (SaaS) platforms play a key role within this cloud environment. SaaS, as a part of the cloud offer, allows to the software providers to deploy and manage their own applications in the clouds in a subscription basis. The problem with the current SaaS offers is the lack of openness of in their platforms and the need for learning a whole new paradigm when trying to initiate in the SaaS market. Big players, such as: Amazon, Google or Microsoft, offer their proprietary SaaS solutions. Another consideration is the amount of current Web applications that need to be re-engineered into this cloud paradigm. This research work aims to reduce the effort required to enter into the SaaS market by presenting an architecture based on open source components for developing, deploying and managing SaaS applications.**

Keywords - cloud computing; software-as-a-service; software architecture; open source.

I. INTRODUCTION

Software-as-a-Service (SaaS) has become the new buzz-word around software industry. From a successful business such as Salesforce.com towards new SaaS software architectures with legacy solutions [3], SaaS solutions have been converted into state-of-the-art technology. In spite of the growth of this industry, there is a lack of established software architectures that enable the delivery of business applications as services. Big players (e.g., Microsoft, Google, Amazon) have developed their own SaaS infrastructure in order to deliver their next-generation software applications. As the number and scale of cloud-computing systems continues to grow, significant research is required to determine the strategy towards the goal for making future cloud computing platforms successful. Currently, most cloud-computing offerings are either proprietary or depend on software that is not amenable to experimentation or instrumentation [1][3][4].

New Internet-enabled platforms have appeared, thus enabling open collaboration and creation. These platforms represent a new way of delivering software applications [2][3]. While the practice of outsourcing business functions such as payroll has been around for decades, its realization as an online software service has until recently became popular. In the online service model, the provider develops an application and also operates the servers that host it. Customers access the application over the Internet using industry-standard browsers or Web Services clients [4][6]. Online software delivery is now conceived and defined as Software-as-Service (SaaS). SaaS has become a well established phenomenon in some areas of enterprise IT. It is growing into a mainstream option for software-based solutions and this will impact most of the enterprise IT departments over the next three years [9]. Chou [5] declares that SaaS is the next step in the software industry, not because it is a "cool idea", but because it fundamentally alters the economics of software.

A wide range of online applications, including e-mail, human resources, business analytics, customer relationship and enterprise planning, are available [6]. According to Gartner [8], the SaaS market will be growing in the next years, by 2009 100% of tier 1 consulting firms will have a SaaS practice and by 2011, 25% of new business software will be delivered as SaaS. Also, IDC estimates customers spending on SaaS solutions will increase to $14.8 billion by 2011 [11]; two out of three businesses are either buying or considering buying software via the subscription model [10] and McKinsey reports that the proportion of CIOs considering adoption of SaaS applications in the coming year has grown from 38% a year to 61% [7]. With previous business facts, is possible to realize the importance and quantity of software that will be delivered throughout SaaS environments.

Unfortunately, several SaaS providers offer their own architecture and their own implementation requirements. Salesforce.com, for example, provides the Force.com development platform and it uses a proprietary development model (custom classes and user interfaces) for building SaaS applications. Furthermore, transition from current Web applications or Application Service Providers (ASPs) solutions to this development model is not a trivial task. Concha, et al. [3] and Espadas, et al. [4] identify a number of steps about transitioning from an ASP solution to a SaaS implementation:

- Current ASPs define a single static revenue models (e.g., embedded & hard-coded within the application implementation). When the dynamic nature of markets asks ASPs to modify their revenue model, ASPs are not able to change it in a cost-effective way, mainly because the revenue model is hard-coded into the application.

 ISBN: 978-1-61208-106-9

- Traditional ASPs provide a portal mechanism for accessing their applications. The current implementation of ASP only supports the notion of one service provider. This is the host platform it-self. The benefits of shifting to a multi-provider approach include an easy integration with associates that complement the platform administrator. Migrating to a multiple provider with multiple e-services also provides the ability to deploy and manage independent sets of applications.
- Presently, ASP services are designed, developed and deployed as Web applications. They are managed by the platform through a Web container and there is no other support for them (such as: billing, monitoring, customization, etc.). In other words, we could see SaaS applications as desktop applications running within an operating system.

This research work addresses these issues, by proposing an open architecture for achieving an implementation capable of deploying applications over the Internet on the service premise. This paper is structured as follows: Section II describes the software architecture and core technologies of the SaaS platform; Section III outlines a set of business services that support SaaS applications; Section IV defines a SaaS application and its components; Section V explains the SaaS core application that is used to access to the platform and Sections VI and VII describe multitenant implementation of applications and subscriptions.

II. SAAS ARCHITECTURE

The architecture bases the communication layer on a Service Oriented Architecture (SOA) that supports techniques for constructing reliable services on cloud computing infrastructures [15].

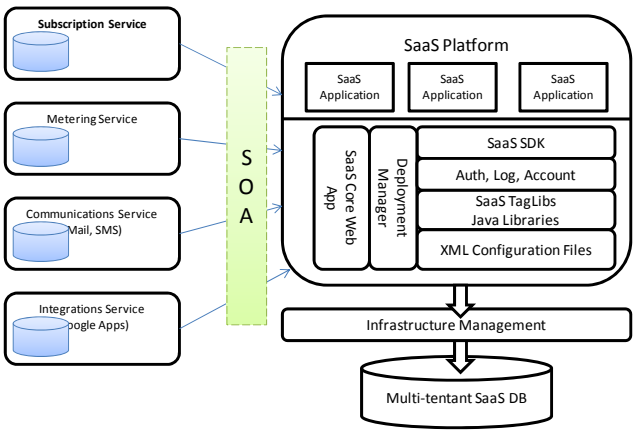


Figure 1. SaaS architecture

The SaaS platform is composed of several components that allow the deployment of applications as services (Fig. 1). Each component is integrated in an Apache Tomcat container as a Web application (.war), a packaged library (.jar) or a business services (Web application + Web Services). A '*service application*' is defined as the application that will be delivered as a service to the customers. Each service application is deployed as a common Web application within the Tomcat container and it manages its own resources, such as data sources, libraries, and views. The main difference with common Web deployments is about how the SaaS components manage and interact with these Web applications. The main interaction point of the service application with the platform is done through a SaaS Application Programming Interface (API). The SaaS API provides the common libraries that are used by the applications to access the basic SaaS services, such as: authentication, account information, public resources and so on. In the view layer, the platform offers components (SaaS Tag Libraries) for an easy integration with the SaaS context (such as: public/private menus, templates, layouts). The Deployment Manager is a listener component that configures each application according to its configuration file (*appService.xml*). Every time a Web application is deployed within the Tomcat container, the Deployment Manager reads the configuration file and analyzes the application code for detect updated or new modules, security roles or deployment changes. The access point to the SaaS platform is the SaaS Core Web Application (SCWA). This component is a Web application that is used to access to all other applications and components. SCWA is in charge of loading common resources and views, such as security context, authenticated user, view filters, etc.

TABLE I. OPEN SAAS PLATFORM TECHNOLOGIES

Requirement	**Technology**
Language Platform	J2EE (Java 1.6)
Web Container	Apache Tomcat 6
Web Framework	Struts 2
Web Services	Apache Axis2
Dependency Injection	Spring 2
Dependency Injection + Web Services integration	WSO2
Multi-tenancy Layer	JoSQL + Java Annotations
Persistence Layer	Hibernate 3 and Java Persistence API (JPA)
Database Management	MySQL 5

As Table I outlines, the core technologies of the SaaS implementation are open source projects. In Figure 1 it is possible to find a set of business services that are consumed through a platform. These business services were designed, developed and deployed by following a Service Oriented Architecture (SOA) design in order to be completely decoupled to the SaaS platform. Each business component exposes a set of Web Services that can be consumed through the platform (or even others platforms) as a client. But this schema can be bidirectional; a business component can be a client of the platform as well. The implementation of these business services will be explained in a further section.

 ISBN: 978-1-61208-106-9

III. BUSINESS SERVICES

A set of support business services is available for service applications. As such, the SaaS applications do not implement code for these mechanisms as they are provided by the platform. The implemented services are:

- *Metering & monitoring.* SaaS platform provides automatic and non-intrusive support for metering applications and tenant-based monitoring.
- *Mailing.* A component for sending/managing electronic mail within applications without complex configuration and programming.
- *Application customization.* The customization component allows the subscriber to customize their own data by adding fields to their business objects (e.g., contact, lead, bill, etc.). By adding custom fields to business object it is possible to generate personalized capture and search forms and to create filters for these custom properties.

Each business component is developed as a Web application, but it exposes a set of Web services through WSO2 framework [16], which integrates web services deployed through Apache Axis2 and dependency injection with Spring 2. Each business component application implements its own Web services and they are referenced in the *applicationContext.xml* Spring file. In this way, any application in the platform can expose its own Web services through simple classes, without having to implement complex mechanism to generate WSDL documents. Same implementations were followed for other business services (e.g., metering, subscriptions, customization). Though, the business services implement other functionalities for their own management and configuration. For example, Mail Service application offers the possibility for configure manage their mail queue and the providers' mail accounts.

IV. SAAS DEVELOPMENT & DEPLOYMENT

A SaaS application is a Web application deployed within the SaaS platform with a particular configuration. A service application is a set of Web components that can be seen as a whole software application. It provides a set of functions separated by modules that can be deployed on demand into a SaaS platform. In a simple way a service application only has:

- *Views.* All the screens and forms that the user can interact with.
- *Business Logic.* Code for actions, business logic and data source accesses.
- *Configuration files.* XML or properties files.
- *Database.* Storage for application data; logically separated for each subscriber.

That is, the SaaS application must not implement code for authentication and authorization, application metering, customization, etc, because they must be provided by the SaaS platform (as described in Section III). In the SaaS platform, the applications' services are developed and deployed as common Web applications but with specific features and configurations that are interpreted by the platform in order to create a SaaS execution environment. The common frameworks and libraries used to develop SaaS applications in the platform are the same as outlined in Table 1. As stated, each application manages its own data sources (e.g., databases, Web services, etc.). The SaaS platform automatically detects configuration such as: business modules, roles, menus, permissions, etc. Once the Web application is deployed and configured, it can be offered as a SaaS solution to multiple customers through a subscription basis. The principal configuration file for deploying a Web application as a SaaS application is the *appService.xml* file. It defines the principal information of a SaaS application.

```
<?xml version="1.0" encoding="UTF-8"?>
<appService>
        <name>Contact Manager</name>
        <label>menu.contactapp</label>
        <version>1.0</version>
        <description>...</description>
        <defaultProvider>TGHEWFS</defaultProvider>
<Role name="manager" description="...">
<Menu>
<MenuItem label="Contacts" path="/contacts/view.action"/>
<MenuItem label="Configure" path="/config/view.action"/>
</Menu>
</Role>
<Role name="user" description="...">
<Menu>
<MenuItem label="Contacts" path="/contacts/view.action"/>
</Menu>
</Role>
</appService>
```

In the last snippet, the XML tag *appService* encloses a set of attributes for the application, such as name, label, description and the default provider which owns the application. The *Role* tag specifies available roles for the application permissions. These roles are updated in the platform database when the platform is initialized in the application server. Within these role tags it is possible to specify application's menus that are presented when the authenticated user has such role. A SaaS application is packaged as a *.war* Java component.

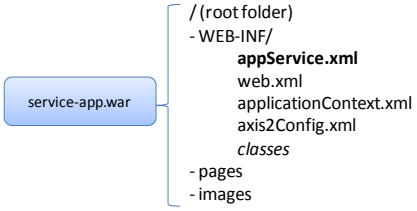


Figure 2. Folder structure of a SaaS application

Figure 2 shows the structure followed by any Web application that is deployed as a SaaS application. This structure shows the location of *appService.xml* file in order to be recognized as a SaaS application by the platform.

 ISBN: 978-1-61208-106-9

The following steps are performed during each SaaS application initialization:

1. A platform component called Deployment Manager reads the *appService.xml* file. This component retrieves information from the service, such as name, version, etc. It inserts or updates the application information in the platform database.
2. Deployment Manager reads the *appService.xml* to create or update the application roles.
3. Deployment Manager inspects the application code for Action classes. This inspection looks up all Java packages that end with '*.actions*' and the classes whose name ends with '*Action*'. For example:
   - *com.myapplication.actions.ContactsAction*
   - *com.myotherapplication.actions.SomeOtherAction*

   These action classes will be inserted or updated in the platform database as modules.
4. Platform inspects each method of a Action class (module). With the use of the *@SaaSFunction* Java annotation it is possible to define functions for each module. This function declaration allows having a method-level granularity about restricted access for each application role.
5. Both modules and functions are synchronized with the platform database.

V. SAAS CORE WEB APPLICATION

The access point for the whole SaaS platform and its deployed applications is known as SaaS Core Web App (SCWA). This component is a Web application with specific characteristics for managing tenant-based authentication, security and control access lists. Each user belongs to one or more subscriber or tenant (these terms will be used indistinctly). Once the user has been authenticated through an email and password, SCWA links the user to its subscriber ID. If the user belongs to two or more subscribers, a selection screen is displayed to select which to work with. After that, SCWA searches for the user within an Access Control List (ACL) to retrieve its permissions for that subscriber and for the SaaS applications that the subscriber has contracted. Then SCWA forms a session cookie with all this information and stores it within the user session. In this way, each user is linked to this tenant-based information and all subsequent requests are identified by this session cookie. However, it is necessary to retrieve this information from several service applications, even when deployed in different machines over a cluster. In order to achieve this, all Web applications should have access to the SCWA and should retrieve the cookies from it. The tenant-based information is stored in the SCWA context session and the rest of the applications can access it through the following mechanism:

```
public static UserVO getAuthenticatedUser() throws
NotAuthenticatedUserException {
        HttpServletRequest request =
ServletActionContext.getRequest();
        String SAASADMIN_SESSIONID =
getCookieValue(request,AuthConstants.SIDEL_SESSION_ID);
        ServletContext contextAuth =
request.getSession().getServletContext();
        UserVO authUser =
getUserFromAdminContext(contextAuth,SAASADMIN_SESSIONID
);
if (authUser==null){throw new NotAuthenticatedUserException();}
        return authUser; }
private static UserVO getUserFromAdminContext(ServletContext
context, String ssosessionid) {
        ServletContext sidelcontext =
context.getContext(SAAS_CORE_APP );
         Hashtable<String, UserVO> shareddata =
(Hashtable<String, UserVO>)sidelcontext.getAttribute(
AuthConstants.SAAS_USERS );
if (shareddata!=null && ssosessionid!=null) {
            // get the right User using the sessionid
            return (UserVO)shareddata.get(ssosessionid);}
        else  return null;
}
```

The static method *getAuthenticatedUser()* can be called from any application and it retrieves the session cookie of the authenticated user from the SCWA context (represented by *SAAS_CORE_APP* variable). *UserVO* is the value object that holds information about the subscriber and the authenticated user.

VI. PERSISTENCE MULTITENANT IMPLEMENTATION

There are different mechanisms for supporting multi-tenancy. The applications services deployed within our SaaS platform implements a *Shared Database - Shared Schemas* mechanism [12][13][14], by separating (logically) the data corresponding to each tenant with a subscriber ID field in the database's tables. This shared schema approach has the lowest hardware and backup costs, because it allows serving the largest number of tenants per database server [14]. As described, each service application implements its own database, separating the multitenant information with a subscriber ID key. An example of a multitenant data model is showed in Figure 3:

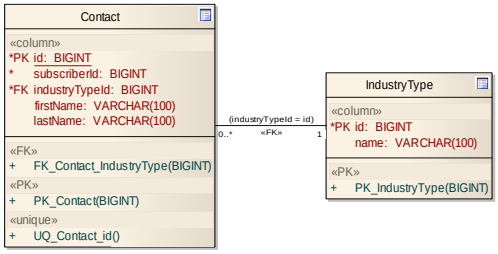


Figure 3. Shared schema for a SaaS application

In Figure 3, It is depicted a shared database scheme for a Contact Manager SaaS application. The core entity is the Contact and each subscriber manages its own set of contacts. As Figure 3 shows, this table has the '*subscriberId*' field; it means that Contact is a tenant-based object. When users log in to the SaaS platform and access to the Contact Manager application, they are allowed to access only to the contacts of their subscribers. As described in the previous section, this subscriber identifier can be retrieved from the SCWA context by any other service application, in this case the Contact Manager. Within the business logic of the application it is possible to retrieve this subscriber ID

 ISBN: 978-1-61208-106-9

and filter the contacts in the persistence layer. The SaaS platform uses an object-oriented mechanism for multi-tenancy which is implemented in the application side and it is called Multi-Tenant Persistence layer. This layer uses JoSQL [17], a LINQ-like [18] technology for perform SQL-like queries over collections and the ability of Struts2 to create Aspect-Oriented interceptors that allows to separate in a logical way the information of each subscriber, supposing the need to retrieve the contacts from a given subscriber. The persistent layer is based on Object Relational Mapping technologies (JPA + Hibernate). We can use a simple object-oriented query to do that:

```
// JPA query in the persistence layer
String sql = "SELECT contact FROM Contact contact"
Query query =  em.createQuery(sql);
return query.getResultList();
```

Simple as is, it is important to notice that there is no filter by subscriber in the query sentence. The persistence layer will return a set of 'Contact' objects. By using the interceptor feature of Struts2 is possible to pre-process these results before they can be accessed by the presentation layer. Within a Struts2 action we can declare an annotated property:

```
@Multitenant(attribute="subscriberId")
List<Contact> contacts = //get contacts from persistence layer
```

The previous code declares that this particular list of objects will be filtered before they are accessible from another component of the application (a Java Server Page view for example). This pre-processing implementation is achieved by setting a Struts2 interceptor in the call stack. This interceptor can access to the invocation action:

```
Object action = invocation.getProxy().getAction();
//getting the subscriber ID from the authentication context
long subscriberId = Auth.getSubscriberId();
for (Field field : clazz.getDeclaredFields() ){
        if (field.isAnnotationPresent(Multitenant.class)){
        Multitenant filter =
(Multitenant)field.getAnnotation(Multitenant.class);
String attribute = filter.getAttribute();
String property = field.getName() ;
Object objList = BeanUtils.getProperty(action, property);    String
className = getClassName( objList );
Query q = new Query ();// create and perform a query over the  list
q.parse("SELECT * FROM "+className+" WHERE "+attribute+"
= "+subscriberId);
QueryResults qr = q.execute (list);
List newList = qr.getResults();
//setting back the filtered list by tenant
BeanUtils.setProperty(action,property,newList);
}}
```

In the example, the 'Contacts' list will be reduced to only the objects which their "*subscriberId*" property matches with the authenticated subscriber. With this mechanism it is possible to create multi-tenant pre-processing behavior within the SaaS applications. In fact, it is feasible to create a transparent support for multi-tenant persistence without affecting the on-premise applications.

VII. MULTITENANT SUBSCRIPTIONS

Basically, the subscription service is a Web application. It uses the same open technologies as the SaaS platform (see Table 1). Its architecture defines a set of components for the subscription management. The storage layer is composed of the multi-tenant database and the logical persistence separation. Different types of subscriptions are handled by a component called Subscription Type Management. As each provider can define its resources for their applications, the Restriction Management is in charge of managing these resources and linked them to a Restriction definition. The Resource Management Remote layer performs access to distributed resource managers from different and heterogeneous sources. This is an important concept of the subscription component because it has the ability to manage distributed resources in different scenarios, either for on-premise ASPs applications or SaaS solutions. As such, the subscription component uses a distributed architecture based on SOA in order to be adaptable to several scenarios. Each entity can define its own resource managers as Web services and these can be consumed for the subscription component dynamically. With this approach it is not only possible to have applications using the subscription component as well as entire platforms consuming the web services. These resources can be any type of accountable and billable resources such as persistent rows (e.g., contacts, leads, bills, surveys, etc.) or hardware (e.g., CPU cycles, bandwidth, storage, etc.). A Resource Manager interface defines a set of methods to be implemented by SaaS services. It defines methods that can be called externally due to the fact that each resource manager implementation is exposed as a Web Service in order to be consumed for the subscription service. This approach allows the dynamic integration of heterogeneous providers. Resource Manager registration is performed when a provider (e.g., a subscriber per se) defines a Restriction for each resource. Therefore, a Restriction (in the Subscription component side) will access its Resource Manager (in the application side). External providers can define their restrictions by using the Subscription Web Application front-end and this is done through the Restriction Management internal component. Internal applications of the SaaS platform are automatically analyzed by discovering their Resource Managers. The Web Services implementation in the SaaS platform is done with Apache Axis2, Spring2 and the integration library between them called WSO2 [16]. Each SaaS application implements its own resource manager, which is referenced in the *applicationContext.xml* Spring file. The subscription service implements multi-tenancy subscriptions with logical separations in its database, by using a subscriber ID field, in order to manage multiple subscribers and subscriptions. A subscriber can be any entity that has resources to bill or to consume. A subscription is a tree-relationship entity composed by two subscribers (client and provider) and a SaaS application. Therefore, we can say that "*subscriber A is subscribed to the Contact Manager SaaS application provided by subscriber B through the subscription number 1234*", as depicted in Figure 4:

 ISBN: 978-1-61208-106-9

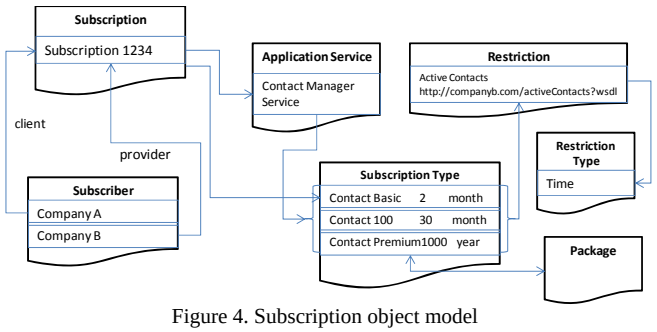


Figure 4. Subscription object model

Each provider can deploy a set of applications and their corresponding subscription types. Subscription Types are the billable plans belonging to a specific SaaS application and they are applicable to a subscription. Packages are used to combine two or more subscription types. As Figure 4 shows, the Contact Manager service defines three subscription types, depending on the contact generation rate. The subscription '1234' defines the 'Contact Basic' subscription type. As such, the 'Contact Basic' plan is going to charge 2 currency units for every active contact with a monthly fee basis. Then, an 'Active Contacts' is defined as a resource to be billed. In order to manage these resources, a subscription type has a Restriction, which is a condition to be monitored by the subscription component and it allows managing the resources' accountability. In the previous example, the 'Active Contacts' resource is defined in the Restriction table, and it defines the source of this resource (as a WSDL end-point). Subscription service implements a Resource Lookup and Invoice Generation mechanisms which use the WSDL end-points in order to gather specific information about billed resources status.

VIII. CONCLUSION

A Software-as-a-Service (SaaS) platform has been described and its implementation on open source technologies. This platform implements a set of business services and components to deploy Web applications and manage them as SaaS solutions. These SaaS applications use shared database schema in order to implement multi-tenancy mechanisms by logically separating their data for each subscriber. The authors' SaaS platform provides an easy transition from traditional ASP applications with single provider approach to more complex scenarios for the next generation of Internet-based services. As such, this platform and its business services are currently used for deploying industry-class SaaS solutions in real production environments.

ACKNOWLEDGMENT

The research presented in this document is a contribution for the "*Rapid Product Realization for Developing Markets Using Emerging Technologies*" Research Chair, ITESM, Campus Monterrey, and for the "*Design of Mechatronics Products*" and "*New Business Models for the Knowledge Economy*" Research Chairs, ITESM, Campus Ciudad de México.

REFERENCES

[1] Nurmi, D.; Wolski, R.; Grzegorczyk, C.; Obertelli, G.; Soman, S.; Youseff, L., and Zagorodnov, D. "*The Eucalyptus Open-Source Cloud-Computing System*". May 2009. CCGRID '09. 9th IEEE/ACM International Symposium on Cluster Computing and the Grid, pp. 90-100.

[2] Molina, A.; Mejía R..; Galeano N.; Najera T., and Velandia M**.** "*The HUB as an Enabling IT Strategy to Achieve Smart Organizations*". Chapter III in Integration of ICT in Smart Organizations, Istvan Mezgar (Editor), Idea Group Publishing, 2006, pp. 64-95.

[3] Concha, D.; Espadas, J.; Romero, D., and Molina, A. "*The e-HUB Evolution: From a Custom Software Architecture to a Software-as-a-Service Implementation*". 2010. Journal Of Computers In Industry. Volume 61, Issue 2, pp. 145-151.

[4] Espadas, J.; Concha, D. and Molina, A. "*Application Development over Software-as-a-Service Platforms*". 2008. The Third International Conference on Software Engineering Advances ICSEA 2008, pp. 97-104.

[5] Chou, T. "*The End of Software*". Sams Publishing, USA, 2005.

[6] Jacobs, J. "*Enterprise software as service*". July 2005. Queue, Volume 3, Issue 6, pp. 36-42.

[7] Dubey, A. McKinsey. Panel at the SIIA OnDemand Summit. San Jose, CA. November 8, 2006.

[8] Gartner Research. "*Predicts 2007: Software as a Service Provides a Viable Delivery Model*". 2006 Gartner, Inc.

[9] Natis, Y. V. "*Introducing SaaS-Enabled Application Platforms: Features, Roles and Futures*". 2007 Gartner, Inc.

[10] Pring, B.; Bona, A.; Holincheck, J.; Cantara, M. and Natis, Y. V. "*Predicts 2008: SaaS Gathers Momentum and Impact*". January 2008. Gartner Inc.

[11] Wolde, E. Research Analyst, IDC. August 2007.

[12] Aulbach, S.; Grust, T.; Jacobs, D.; Kemper, A., and Rittinger, J. "*Multi-tenant databases for software as a service: schema-mapping techniques*". June 2008. SIGMOD '08: Proceedings of the 2008 ACM SIGMOD international conference on Management of data, pp. 1195-1206.

[13] Mietzner, R.; Metzger, A.; Leymann, F., and Pohl, K. "*Variability modeling to support customization and deployment of multi-tenant-aware Software as a Service applications*". May 2009. PESOS '09: Proceedings of the 2009 ICSE Workshop on Principles of Engineering Service Oriented Systems, pp. 18-25.

[14] Frederick C.; Gianpaolo C., and Roger W. (June 2006). "*Multi-Tenant Data Architecture*". MSDN Library. Microsoft Corporation. http://msdn.microsoft.com/en-us/library/aa479086.aspx. Last access at January 2009.

[15] Sedayao, J. (November, 2008). "*Implementing and operating an internet scale distributed application using service oriented architecture principles and cloud computing infrastructure*". iiWAS '08: Proceedings of the 10th International Conference on Information Integration and Web-based Applications & Services, pp. 417-421.

[16] Mathew, T. (February, 2008). "*Hello World with WSO2 WSF/Spring*". WSO2 - The Developer Portal for SOA. http://wso2.org/library/3208. Last access at June 2009.

[17] Haines, S. (Sept, 2005). "*JoSQL - SQL for Java Objects*". http://www.informit.com/guides/content.aspx?g=java&seqNum=230. Last access at July 2009.

[18] Kan, W. and Yujun, Z. (2009). "*Using LINQ as an instructional bridge between object-oriented and database programming*". ICCSE '09. 4th International Conference on Computer Science & Education, 2009, pp. 1464 - 1468

 ISBN: 978-1-61208-106-9

Behaviour-inspired Data Management in the Cloud

Dariusz Król, Renata Słota, Włodzimierz Funika
Institute of Computer Science
AGH - University of Science and Technology
al. Mickiewicza 30, 30-059 Krakow, Poland
{dkrol, rena, funika}@agh.edu.pl

***Abstract* — Open source cloud computing solutions are still not mature enough to handle data-intensive applications, e.g., scientific simulations. Thus, it is crucial to propose appropriate algorithms and approaches to the data management problem in order to adjust cloud-based infrastructures to scientific community requirements. This paper presents an approach inspired by observation of the cloud user behaviour: the intensity of data access operations, their nature, etc. We also describe how the proposed approach influences the architecture of a typical cloud solution and how it can be implemented based on the Eucalyptus system which is a successful open source cloud solution.**

Keywords- cloud computing; data management; monitoring.

I. INTRODUCTION

Cloud computing is arguably the most popular buzzword in the tech world today. It promises to reduce the total cost of maintenance of an IT infrastructure with providing better scalability and reliability at the same time. Apart from "unlimited" computational power, the Cloud provides also "unlimited" storage capacity which can be accessed from every device which is connected to the Internet. Therefore, this is not a surprise that many commercial companies along with academic facilities are very interested in this paradigm. As with other paradigms, various research centers test it in a variety of ways in order to unveil its strengths and weaknesses. What makes the Cloud computing special in comparison to other, more academic-related approaches to distributed computing, e.g., Grid computing, is the support and investment made by the largest IT companies, e.g., Google, IBM, Microsoft and many others. These million dollars investments can be treated as a good omen that Cloud computing can be widely adopted and will not disappear after few years.

Today, cloud computing solutions, especially the open source ones, are not mature enough in terms of storage capabilities to handle data-intensive applications which would like to store results in the cloud-based storage. One of the issue is lack of adaptation of data management strategy, e.g., to changing user requirements or location from where the user access the cloud storage. Each of these aspects can imply the access time to data especially when considering geographically distributed resources which constitute a single cloud installation. Therefore, we propose a novel approach, based on autonomic systems (similar to situation-aware systems - [1]) and behaviour observation whose main goal is to adapt data location to the user needs which will result in decreasing access time and higher utilization factor of resources. We introduce a "Usage profile" concept which describes a piece of data stored in the cloud storage. The usage profile contains information how the described data is used by cloud clients. To create such a profile, operations related to data storage performed by cloud users are monitored and analyzed. The approach is designed to be an additional element of the cloud installation rather than being mandatory, which is invisible from the user point of view but can positively influence the storage performance.

The commercial clouds, e.g., Amazon Elastic Compute Cloud (EC2) [2] which it is an Infrastructure-as-a-Service system, which means it allows to manage a computational environment consisted of virtual machines, cannot be easily studied due to closed source code, thus in this paper they will not be taken into consideration. The rest of the paper is organized as follows:

- in Section II, a number of the existing cloud solutions are presented,
- Section III describes the data management algorithm which is based on the behaviour analysis,
- parameters of the usage profile along with behaviour which is analyzed in order to create the usage profiles are described in Section IV,
- a prototype implementation of the algorithm is presented in Section V,
- conclusions along with the future directions of the research are provided in Section VI.

II. EXISTING OPEN-SOURCE CLOUD ENVIRONMENT

Cloud computing has been already widely adopted by various commercial companies and academic centers. While many commercial companies develop their own solutions, e.g., Amazon EC2, Microsoft Azure [3] or Google AppEngine [4], others use and invest in open source solutions which are especially well suited for situations where the environment has to be adapted to some specific requirements. This feature is very important for scientific community which would like to implement new concepts and approaches to optimize access time or other parameters. Thus in the presented research only open source solutions were taken into consideration. In this section, we describe three well known "Infrastructure as a Service" environments: Eucalyptus, Nimbus and OpenNebula.

 ISBN: 978-1-61208-106-9

A. *Eucalyptus*

Eucalyptus system [5] is an example of an open source project which became very popular outside the scientific community and is exploited by many commercial companies to create their own private clouds. It was started as a research project in the Computer Science Department at the University of California, Santa Barbara in 2007 and today is often treated as a model solution for providing infrastructure as a service. Eucalyptus aims at providing an open source counterpart of the Amazon EC2 cloud in terms of interface and available functionality.

Each Eucalyptus installation consists of a few loosely coupled components each of which can run on a separate physical machine to increase scalability. The frontend of such a cloud is "Cloud controller" element which is an access point to the virtual machines related features. While "Cloud controller" is responsible for computation, the "Walrus" component is responsible for data storage. It allows storing virtual machine images along with any other files which are divided into a hierarchy of *buckets* and can be treated as the Amazon Simple Storage Service (S3) counterpart in the Eucalyptus system. Amazon S3 is a Cloud storage service which allows storing any type of data in form of files in a number of buckets (each with a unique name within a bucket) using a simple API, i.e., *put, get, list, del* operations are supported. Each virtual machine is run on a physical host which is controlled by the "Node controller" element. A group of nodes can be gathered into a cluster which exposes a single access point, namely "Cluster controller" from the virtual machine management side and "Storage controller" from the virtual machine images repository side.

Eucalyptus is based on the Java technology stack and its source code is freely accessible and can be modified as necessary. To mention a few, the current implementation uses web services (Apache Axis [6]) to expose the provided functionality to the external clients, and exposes a web-based user interface developed with the Google Web Toolkit (GWT) [7]. It also supports the Xen [8] and KVM [9] hypervisors to run virtual machines on the supervised resources.

The open version of the Eucalyptus system stores data into a single directory on the host on which the Walrus component is installed. Therefore, the only way to distribute the data is to exploit a distributed file system, e.g., Lustre [10], which will be mounted to the directory used by the Eucalyptus installation. This can be unfortunately not enough especially when the application is running.

B. *Nimbus*

Nimbus [11] is a toolkit for turning a cluster into an IaaS cloud computing solution. It is developed by the Globus Alliance [12]. A Nimbus client can lease remote resources by deploying virtual machines on these resources and configure them to fulfil the user requirements. What makes it attractive is the support of a communication interface known from the Grid computing, namely Web Services Resource Framework [13]. As in other popular solutions, Nimbus provides an Amazon EC2 compatible interface for cloud clients, which is de facto a standard of IaaS environment due to its wide adoption in a number of solutions.

A Nimbus installation consists of a number of loosely coupled elements. The center point of the Nimbus architecture is the "Workspace service" component which is a coordinator of the whole installation. It is invoked through different remote protocol frontends, e.g., WSRF or EC2 – compatible services. Another important component is "Workspace resource manager" which runs on each host within the Cloud and is responsible for controlling a hypervisor on the host machine. The current version fully supports Xen hypervisor and most of the operations on the KVM hypervisor. It is also worth of mentioning that Nimbus installation can be easily connected to a public commercial cloud, e.g., Amazon EC2 in order to achieve even greater computer power when the in-house infrastructure is not enough.

In terms of data management, the Nimbus project is limited to the virtual machine image repository. There is no component which would provide functionality similar to the Amazon S3. The user can only upload virtual machine images to the Nimbus cloud and store data stemming from computation on storage devices connected directly to a virtual machine.

The Nimbus project is based on open source tools and frameworks, e.g., Apache Axis, the Spring framework [14] or JavaDB [15]. Therefore, everyone can download its sources from a public repository and modify its functionality as desired.

C. *OpenNebula*

OpenNebula [16] is a Virtual Infrastructure Manager for building cloud infrastructures based on Xen, KVM and VMWare virtualization platforms [17]. It was designed and developed as part of the EU project RESERVOIR [18], whose main goal is to provide open source technologies to enable deployment and management of complete IT services across different administrative domains. OpenNebula aims to overcome shortcomings of existing virtual infrastructure solutions, e.g., inability to scale to external clouds, a limited choice of interfaces with the existing storage and network management solutions, few preconfigured placement policies or lack of support for scheduling, deploying and configuring groups of virtual machines (apart from the VMWare vApp solution [19]). Like other of the presented solutions, OpenNebula is fully open source and its source code can freely be checkout from a public repository.

OpenNebula architecture was designed with modularity feature in mind. Therefore, it can be extended to seamlessly support a new virtualization platform e.g., in terms of virtual image or service managers. For instance, a procedure of setting up a VM disk image consists of well-defined hooks whose implementation can be easily replaced to interface with the third-party software. To manage an OpenNebula installation, the user can use a simple, dedicated command line interface or Amazon EC2 query interface. Therefore, it can be accessed with the tools originally developed to work with the Amazon EC2 cloud.

 ISBN: 978-1-61208-106-9

In terms of storage mechanisms, it is limited to repository of VM images only. The repository can be shared between available nodes with the Network File System (NFS) [20]. It is also possible to take advantage of block devices, e.g., LVM to create snapshots of images in order to decrease time needed to run a new instance of image.

III. BEHAVIOUR-INSPIRED APPROACH TO DATA MANAGEMENT

The most important aspect of the presented approach is its orientation towards each user requirements rather than some global optimization such as equal data distribution among available resources. Our approach treats each user separately by monitoring his/her behaviour related to data storage. The monitoring process is necessary in order to discover the nature of the data automatically, e.g., whether it is read-only or often modified data. With this knowledge, the data can be managed appropriately, i.e., with requirements such as high availability taken into account. Another important feature of the approach which can be deducted from the previous one is its transparency from the user point of view. Thus, it can be applied to any existing solution without any modification required to the user-side code.

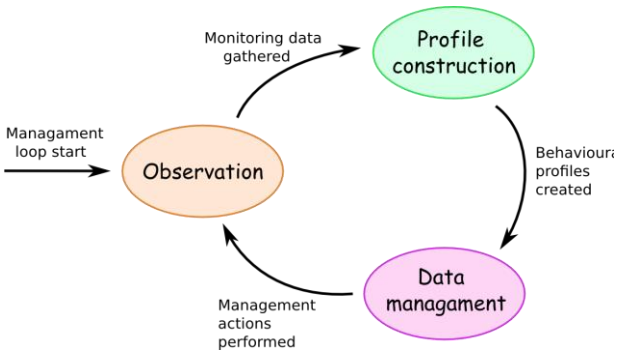


Figure 1: Data management loop.

The structure of the described algorithm is depicted in Figure 1. There are 3 phases included:

- *The observation phase* where the information about the user behaviours are aggregated. It is a start point of the management iteration. Each operation related to the storage, e.g., uploading or downloading files are recorded along with information about the user who performed the operation and a time-stamp.
- *The profile construction phase* is the one where the gathered information about behaviour is analyzed. For each user, a profile which describes how the user accesses each piece of data is created, thus the profile also contains information how each piece of data should be treated.
- *The data management phase* is responsible for modifying the data storage, i.e., applying a dedicated strategy which corresponds to a user profile. Such a strategy can e.g., create many replicas of a piece of data which is read by many users but hardly any one modifies it or it can move the data closer to the user to decrease its access time.

An important feature of the algorithm is the fact that it never ends. There is no stop condition because such a management process may last as long as the cloud is running. Each iteration of the loop results in tuning the storage strategy to the observed user behaviour. However, the historical data is taken into account as well and can influence the storage strategy rather than just be omitted. In fact, its importance to the new strategy is one of the parameters of the algorithm.

Another important aspect of the approach is its influence on the architecture of a cloud solution. The overview of architecture is schematically depicted in Figure 2. To underline the most important components, some simplifications were introduced, e.g., the cloud solution is represented only by "Cloud manager" which is an access point to the cloud infrastructure. "Storage elements" represent physical resources where the data is actually stored. New components are as follows:

- *Monitoring system* is responsible for gathering information about user actions. The most important operations are those related to data storage, e.g., uploading a file or accessing a file by the user. Information about these actions have to be remotely accessible by an external cloud client in a programming language independent way.
- *Behaviour data manager* is the main element of this new approach. It performs the analysis of the user behaviours and creates their profiles. Then, it performs all the necessary actions to adjust the storage strategy to the actual profile. In most cases, these actions will be related to either moving data between storage elements with different physical parameters or managing data replication, e.g., creating new replicas. By combining these two types of operations, we can improve the Quality of Service (QoS) of the cloud storage, e.g., decrease the data access time. It is also possible to apply more sophisticated algorithms for data management as the ones described in [21] and [22]. The communication between “Behaviour data manager” and “Storage elements” is optional. If “Cloud manager” exposes an interface to manage the actual data location, there is no need for “Behaviour data manager” to interact with “Storage elements” directly.
- *Profile knowledge base* is a repository where the historical profile for each piece of data is stored along with record of each performed action. Thus, it can be used by the "Behaviour data manager" to take into account not only the most recent information but also the previous actions.

As we can see the approach can be easily integrated with any cloud solution which can be monitored, i.e., each performed operation related to the storage is registered, and the stored data can be moved between available physical resources, either indirectly with an exposed programming

interface or directly with accessing storage elements and moving raw data. These requirements are rather easy to meet and in the next section we are presenting an example implementation based on a popular open-source cloud solution. It is also worth mentioning that the original source code of such a cloud solution stays untouched.

IV. USAGE PROFILE

The presented approach highly exploits usage profiles which are based on the observation of the user actions. From the performed actions, only those related to the data storage (e.g., uploading or downloading files) are important. In the system a set of profile types is defined along with the actions which are recommended for each of these types. A profile type and an actual profile (which will be described later in this section) correspond to a piece of data, e.g., a single file. Thus, it can be treated as a metadata for the actual data stored in a cloud storage.

Such a profile type contains information which describes how the described data is used. So far a few parameters were defined:

- Category of access frequency which describes whether the object is frequently accessed
- Percentage of the queries relating to read/write operations
- Category number of clients for read/write operations
- Category connection points (e.g., IP addresses) for read/write operations

Based on these parameters a set of a predefined profile types can be created. They should correspond to the well known storage management patterns, e.g., if an object is always read, it should be replicated to multiple physical locations in order to lower the access time.

On the other side, we observed user actions which can be grouped by an object they are related to. As a result, we have got a behaviour profile which contains the following information:

- Frequency of access to an object, e.g., per hour or per iteration of the algorithm loop.
- Number of read/write operations
- Number of different users accessing to the object (separately for read/write operations)
- Number of different places (e.g., IP addresses) from which the object was accessed (separately for read/write operations)

Such a profile is created for each piece of data in the cloud storage (e.g., file) after the first iteration of the algorithm and updated on each subsequent iteration. In each "Data management" phase, for each profile a similarity function to each defined profile type is calculated. Then, the actions related to the most similar profile type are performed.

Therefore, the approach can be easily extended in terms of recognized behaviour, simply by defining new profile types along with related actions.

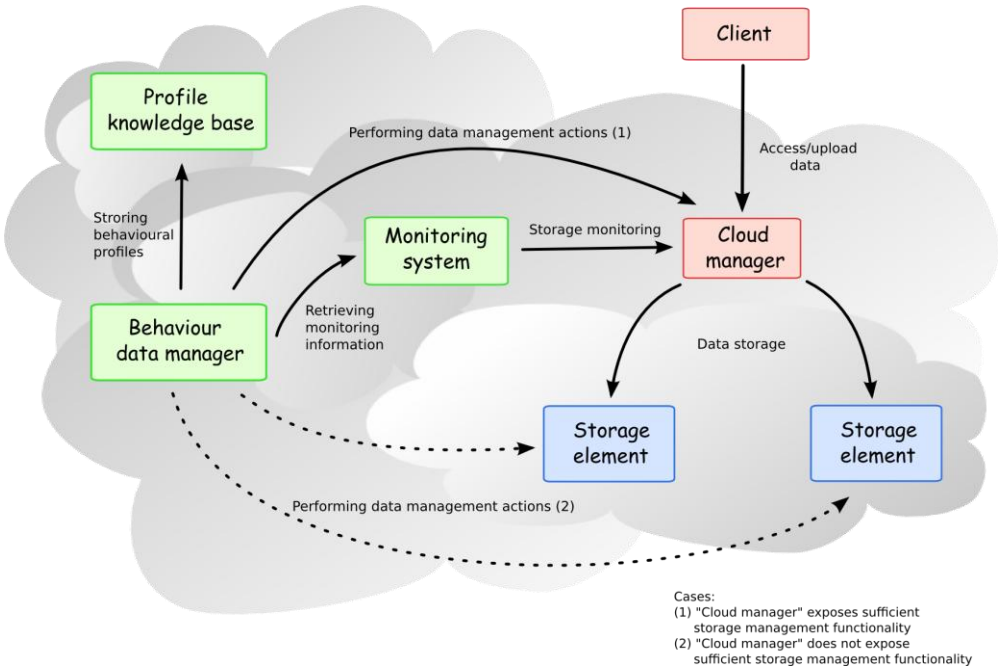


Figure 2: Architecture overview of a cloud solution with "Behaviour data manager" applied.

V. IMPLEMENTATION NOTES

To present a sample implementation of the approach, we chose the Eucalyptus system as a basis. This choice was motivated by a large popularity of Eucalyptus and its

functionality in terms of storage which is very similar to the Amazon S3 offer, a de facto standard in the cloud industry.

The Eucalyptus architecture contains a component called "Walrus" which is responsible for the -storage-related functionality. "Walrus" exposes an API (Application Programming Interface) which consists of methods which create, update, and remove objects and buckets from the cloud storage.

The open version of the Eucalyptus implements the cloud storage as a designated directory on the local filesystem. It is rather a minimalistic solution of the cloud storage, due to very limited ways of data distribution. The possible way is to mount a distributed file system at the designated directory which will transparently distribute data among a number of storage elements. Unfortunately such a solution does not allow to control data manipulation which cannot be accepted in our situation. Therefore, we extended the storage system in Eucalyptus with an ability to store data in several directories instead of one only, each of which can point to a different physical location, e.g., via Network File System (NFS). With this extension, the location of the data can be easily controlled, simply by moving files between directories.

exposes an API to external users for storing data in the cloud. Apart from storing custom data, e.g., results coming from a running simulation, "Walrus" stores two other types of objects. The first one is a VM image which is uploaded by the user and then is run on the Eucalyptus infrastructure. Although, the user communicates with other component, called "Cloud controller", the images are actually stored with "Walrus". The second type of objects is the "Block storage" object. It is used as a mountable partition to store data during a VM run, similarly to a local file system. Moreover, a "Block storage" object can store data between two subsequent runs of a VM and in opposite to a VM virtual disk, the data is not erased after a VM shutdown. Such a partition is stored within the cloud storage with "Walrus". All these three types of objects are stored with "Walrus" on the same rules and thus can be uniformly managed with our system.

The first of the additional elements added to the architecture is a dedicated “Monitoring system”. It consists of “Log analyzer” which periodically reads the Walrus log where each operation related to the storage is recorded and a relational database where information who and which

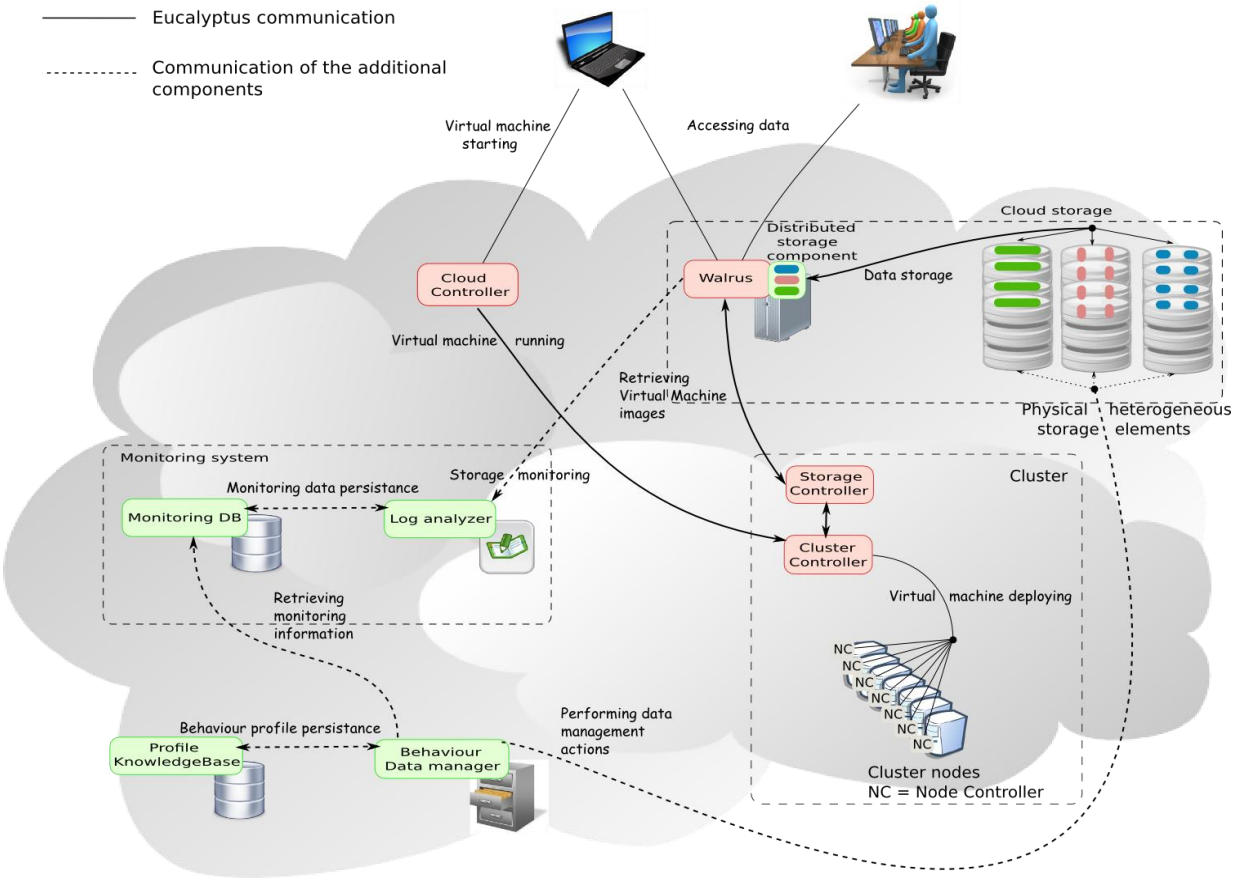


Figure 3: Eucalyptus system architecture extended with “Behaviour Data Manager”, “Profile KnowledgeBase”, “Monitoring system”, and “Distributed storage component”.

As mentioned above there are a few components to add to the Eucalyptus architecture in order to implement the approach under discussion. Such an extended architecture is depicted in Figure 3. There is the "Walrus" component which

operation performed. Thus all the necessary data for profile creation is prepared in a technology neutral form. The second element is "Behaviour Data manager" which is responsible for analysing data written to “Monitoring Database (DB)”. The behaviour of each user related to each stored object is categorized to one of the defined groups, which describes how the object should be treated, e.g., if it is written mostly often by a single user or if it is read by multiple users who are geographically distributed but which is not modified often. This information in form of a “Usage profile” is then stored in "Profile KnowledgeBase" along with its update timestamp. After the categorization phase is over, "Behaviour Data manager" performs actions which are suitable for an assigned usage profile. An action can be as simple as moving an object between directories on the Walrus machine, each of which represents a different type of storage elements, or as complicated as creating many replicas and moving them as close to users as possible. The historical data is taken into account with a weight set in the system configuration.

The whole algorithm is performed periodically in order to adapt the storage strategy to the dynamic environment. The presented implementation is currently in the prototype phase. The monitoring system is finished and tested but the implementation of "Data manager" is still in progress.

VI. CONCLUSIONS AND FUTURE WORK

Although, there are several open-source cloud solutions available today, none of them provides a storage system which would be able to adapt to the user needs automatically. Instead, only basic functionality, e.g., storing VM images or custom objects is supported. The presented approach aims at providing a sophisticated data management which would be flexible enough to be applicable to different cloud solutions and which would manage data according to the user behaviour. The article describes the main assumptions of the approach along with phases of the management process. As an example of its implementation a prototype version of the system based on Eucalyptus is described. Due to the limited functionality of the Eucalyptus system, an extension which provides a real distributed data storage to multiple locations has been implemented.

Since the implementation of the approach is in progress, there is work to be done. The "Data manager" and "Profile database" components are not yet finished. From the conceptual point of view, the approach lacks a well-defined set of behaviour categories along with related actions. However, these categories will be crystallized during real-life tests when the behaviour of the real users will be observed and analyzed. Until then, some preliminary categories will be be defined.

ACKNOWLEDGMENTS

The authors are grateful to prof. Jacek Kitowski for valuable discussions. D. Król thanks to UDA-POKL.04.01.01.01-00-367/08 project for support, R. Słota and W. Funika to the projects A-0938-RT-GC EDA EUSAS Project and POIG.02.03.00-00-007/08-00 “PL-Grid”.

REFERENCES

[1] Han, J.H., Lee, D.H., Kim, H., In, H. P., Chae, H.S., and Eom Y.I., “A situation-aware cross-platform architecture for ubiquitous game”, Computing and Informatics, vol. 28, nr 5, 2009, pp. 619-633.

[2] Amazon Elastic Compute Cloud (Amazon EC2). Amazon Inc., 2008. [on-line: http://aws.amazon.com/ec2, as of June 13, 2010]

[3] Microsoft Windows Azure Platform (Windows Azure). Microsoft, 2010. [on-line: http://www.microsoft.com/windowsazure/, as of June 13, 2010]

[4] Google AppEngine. Google Inc., 2008. [on-line: http://code.google.com/intl/pl-PL/appengine/, as of June 13, 2010]

[5] Eucalyptus Systems Inc.: 2010, [on-line: http://www.eucalyptus.com/, as of July 24, 2010]

[6] Apache Axis website. [on-line: http://ws.apache.org/axis/, as of June 13, 2010]

[7] Google Web Toolkit website. [on-line: http://code.google.com/intl/pl-PL/webtoolkit/, as of June 13, 2010]

[8] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, and A. Warfield, "Xen and the art of virtualization," in SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles. New York, NY, USA: ACM, 2003, pp. 164-177, [on-line: http://dx.doi.org/10.1145/945445.945462 as of June 13, 2010]

[9] Kernel-based Virtual Machine project wiki. [on-line: http://www.linux-kvm.org/page/Main_Page, as of June 13, 2010]

[10] Lustre filesystem wiki. [on-line: http://wiki.lustre.org/index.php/Main_Page, as of June 13, 2010]

[11] Kielmann, T., “Cloud computing with Nimbus”, March 2009, EGEE User Forum/OGF25 & OGF Europe's 2nd International Event.

[12] Globus Alliance website. [on-line: http://www.globus.org/, as of June 13, 2010]

[13] Foster, I., Frey, J., Graham, S., Tuecke, S., Czajkowski, K., and Weerawarana, S., “Modeling Stateful Resources with Web Services”, 2004. [on-line: http://www.ibm.com/developerworks/library/ws-resource/ws-modelingresources.pdf, as of June 13, 2010]

[14] Spring Framework website. [on-line: http://www.springsource.org/, as of June 13, 2010]

[15] Java database website. [on-line: http://developers.sun.com/javadb/, as of June 13, 2010]

[16] B. Sotomayor, R. S. Montero, I. M. Llorente, and I. Foster, "Capacity Leasing in Cloud Systems using the OpenNebula Engine." Cloud Computing and Applications 2008 (CCA08), 2009.

[17] VMware website. [on-line: http://www.vmware.com, as of June 13, 2010]

[18] RESERVOIR project website. [on-line: http://62.149.240.97/, as of June 13, 2010]

[19] VMware Virtual Appliances website, [on-line: http://www.vmware.com/appliances/getting-started/learn/, as of June 13, 2010]

[20] Network File System version 4 protocol specification, [on-line: http://tools.ietf.org/html/rfc3530, as of June 13, 2010]

[21] Slota, R., Nikolow, D., Kuta, M., Kapanowski, M., Skalkowski, K., and Kitowski, J., “Replica Management for National Data Storage”, Proceedings PPAM09, LNCS6068, Springer, 2010, in print.

[22] Slota, R., Nikolow D., Polak, S., Kuta, M., Kapanowski, M. , and Kitowski, J., "Prediction and Load Balancing System for Distributed Storage", Scalable Computing Practice and Experience, 2010, in print.

 ISBN: 978-1-61208-106-9

Semantic Resource Allocation with Historical Data Based Predictions

Jorge Ejarque*, Andras Micsik‡, Raül Sirvent*, Peter Pallinger‡, Laszlo Kovacs‡ and Rosa M. Badia*†
*Grid Computing and Clusters Group - Barcelona Supercomputing Center (BSC), Barcelona, Spain
†Artificial Intelligence Research Institute - Spanish National Research Council (IIIA-CSIC), Barcelona, Spain
‡Distributed Systems Department - Computer and Automation Research Institute
Hungarian Academy of Sciences (MTA SZTAKI), Budapest, Hungary
{jorge.ejarque, raul.sirvent, rosa.m.badia}@bsc.es, {micsik, pallinger, kovacs}@sztaki.hu

***Abstract*—One of the most important issues for Service Providers in Cloud Computing is delivering a good quality of service. This is achieved by means of the adaptation to a changing environment where different failures can occur during the execution of different services and tasks. Some of these failures can be predicted taking into account the information obtained from previous executions. The results of these predictions will help the schedulers to improve the allocation of resources to the different tasks. In this paper, we present a framework which uses semantically enhanced historical data for predicting the behavior of tasks and resources in the system, and allocating the resources according to these predictions.**

***Keywords*-multi-agent, semantics, scheduling, resource allocation, historical data, predictions, grid computing, cloud computing, distributed systems.**

I. INTRODUCTION

The Service-Oriented Computing (SOC) paradigm [1]. relies on the composition of services to build distributed applications by using basic services offered by third parties. Those services are offered by service providers that create their description and their implementation. From the business point of view, the service provider agrees with its customers the Quality of Service (QoS) and level of service through a Service Level Agreement (SLA). The fulfillment or violation of the SLAs indicate the grade of satisfaction of customers with the Service Provider (SP), affecting directly or indirectly to the benefit of these provider. One of the most common SLA violation happens when unexpected events such as failures appear and the system is not able to adapt to this change.

Service adaptation is a current research topic, addressing the automatic reactions to unanticipated events. Adaptation mechanisms usually get active when something already went wrong. However, adaptation can be used also when we anticipate a problem to occur. Prediction mechanisms can help to warn about statistically probable unwanted situations, or we can just calculate the likelihood of certain service parameters. The simple aim is to learn from the past, in order to project events happened in the past to the future. There are several software toolkits supporting similar calculations in the area of data mining, etc.

In this paper, we introduce a generic framework for prediction and adaptation, and describe its application in a specific scenario (Cloud resource scheduling). The basic requirements for such an environment are:

- to provide generic capability of collecting log data about internal events,
- to unify the collected data so that global coherence can be revealed,
- to provide customizable methods for getting predictions based on the collected data,
- to feedback predictions into the realization of the scheduling and adaptation mechanisms.

This framework combines prediction techniques with semantic technologies, which introduces semantic knowledge to the data evaluated by predictors, and multi-agent systems, which introduce the pro-activity and distributed problem solving for increasing scalability, adaptability and self-management of the system. Predictions extracted from the semantic historical data are taken into account by a group of agents for allocating different customer's jobs in the most reliable resources.

The paper is organized as follows: Section II gives an overview of the architecture of our solution; Section III is focused on the implementation of job predictions and their usage in the resource allocation process; Section IV evaluates the framework; Section V compares our proposal with the related work; and Section VI concludes the paper.

II. ARCHITECTURE

Figure 1 shows the architecture of our proposed framework. It depicts a resource allocator distributed across multiple agents based on the Multi-Agent Resource Allocation (MARA) approach [2] whose decision are based on predictions based on historical data. There are two differentiated parts in the architecture: the part which is related to the management of jobs and the part which is in charge of resource provisioning. The job management part allows the customers to make all the actions related to their jobs (submit, cancel, etc), while the resource provisioning part allows the system administrator to add and remove resources.

Both parts are built on top of a JADE agent platform [3]. The platform can be distributed across multiple locations deploying containers on each of them. Moreover, it

 ISBN: 978-1-61208-106-9

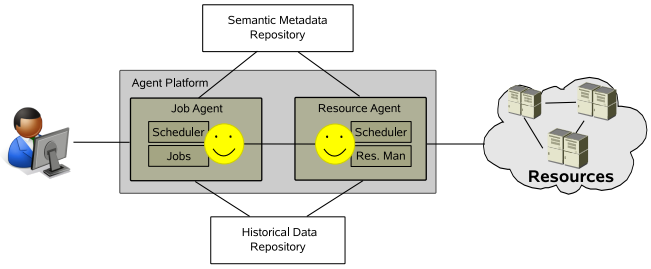


Figure 1. Semantic Resource Allocation with predictions

implements a messaging system, which allows the communication between agents located on different containers. The distributed configuration of the agent platform can improve the scalability of the system because the different parts can be processed in parallel on multiple hosts.

Customers jobs and resources are represented in the system by software agents. Job Agents are in charge of managing the customers jobs and Resource Agents are in charge of managing the providers resources. Moreover, the scheduling of the jobs in the different resources is made by an agreement reached from a negotiation between a Job Agent and different Resource Agents.

Apart from the job management and resource provisioning parts, the system architecture contains a Semantic Metadata Repository (SMR), which contains the semantic resource description registered in the system, and the Historical Data Repository (HDR), which contains semantically annotated logs from system events such as job executions, failures and other monitoring data, which is important to make predictions for current jobs. All the data stored in those semantic repositories is described according to a shared ontology providing a common framework for semantic data. The following paragraphs are focused on the most important part of the architecture.

A. Scenario Ontology

One of the most important issues about semantic technologies is the ontology, which models a common understanding of the concepts used on a scenario of study. In this case, we require an ontology for modeling the system entities such as customers, service providers, jobs and resources and other important data such as resource allocation data for job scheduling and historical data for making predictions.

The Grid Resource Ontology (GRO) [4] provides a model where the most important concepts and entities are described. This ontology provides a definition of the customer jobs and resources but it lacks concepts for describing resource allocation and historical data. Therefore, the GRO has been extended in the scope of the BREIN project [5] in order to cover the mentioned gaps.

The gap of resource allocation data in ontologies was studied in our previous work [6]. Resource requirements have been introduced in the GRO as a set of required abstract *GRO Resources* in the *GRO Task* definition, which models an abstract job. This definition has been also extended with time constraints such as expected duration, deadline, earliest possible start, etc. The scheduling result (assigned resources and time slot) has been introduced on the *GRO Process*, which incarnates the *GRO Task*. Required resources are linked to the task definition as resource sets. A resource set is typically a host, a container of more detailed resource descriptions (disk, CPU, etc.) with resource properties (e.g. size of disk). Furthermore, tasks are also linked to their representing agents and to related business information (such as customer data, service provider, agreed SLA) via the BREIN Business Ontology. These extensions provide the possibility for multi-scope description of job executions merging data about technical capabilities, schedules and business aspects into a single model. It is future work to fully exploit the new capabilities provided by this model. In this paper we provide the first steps in this direction.

GRO Process and *Task* descriptions are also useful for historical data. When a job has ended, the *GRO Process* description contains the result of this job (start time, end time, final status, ...) and the *GRO Task* (abstract job) contains the requested resources, the expected duration and the scheduled start and end times described. The GRO extension for describing historical data also contains classes for describing resources used by each task and resource downtimes. The comparison and evaluation of this data can be used for making predictions about how different types of jobs will run on different resources.

B. System Agents

Our system contains two types of system agents for managing jobs and resources. These agents have been implemented using a Belief-Desire-Intention (BDI) model [7]. For each type of agent, a set of data (Beliefs), goals (Desires) and plans (Intentions) are defined to model the behavior of the agent. Depending on the values of the data, events and the active goals on each moment, the BDI engine decides which plans has to execute the agent for reaching its goals. Our BDI agents have been developed with the Jadex framework [8] used on top of the JADE platform. The following paragraphs give more details about the job and resource agent functionalities and how the BDI model is applied for implementing them.

1) Job Agent (JA): The JA has the main goal of executing jobs in the resources of the system. This execution has different states, some of them indicate that the execution is running correctly and other ones indicate problems in the execution. Depending on the state of the job execution, the JA has to act in a different way. For this reason, the main goal of the JAs has been separated in several subgoals, which will be activated depending on the job status. The activation of subgoals triggers the execution of the plans

 ISBN: 978-1-61208-106-9

to achieve them. For instance, when a new job execution is requested, the JA activates the goal for negotiating a resource allocation. In the *running* state, the JA activates the goal for monitoring the job execution evaluating if the required performance is fulfilled. Finally, if the job is *finished* the JA execute plans for deallocating the resource.

The JA has also to recover itself from status, which can alter the normal execution of the job (*stopped, suspended, non scheduled*). In those situations, the JA will execute plans to resubmit the job or to look for new resources.

2) Resource Agent (RA): The main goal of a RA is the management of resource capabilites for executing the jobs requested by the customer according to resource capabilities and the status and number of jobs assigned to the resources controlled by the agent (jobs *scheduled* and *running*). To provide this main feature, a set of subgoals and plans have been defined to negotiate resource allocations with JAs.

Apart from the negotiation subgoal, RAs contain two subgoals for monitoring the scheduled jobs and running jobs assigned to their resources. These subgoals are in charge of initiating the job execution depending on the planned start time or canceling a job if the deadline has been reached and new jobs are waiting for execution.

Additionally, the RA is also prepared to react to resource failures. The plan for recovering a failure sends a stopped notification message to JAs whose jobs were scheduled at the failed resources. Similarly, it also sends a suspended notification message to JAs whose jobs were already running. JAs treat these notifications according to the plans explained in the JA part.

C. The Historical Data Repository (HDR)

The HDR is a flexible, generic component implemented by SZTAKI to provide facilities for log data collection and on-demand predictions. The HDR is implemented in Java and can be embedded into Java code or can be run as a separate Web Service. In both cases, other software components can send their log data to the HDR, either by our customizable client APIs or via direct calls. Incoming information must be in the format of RDF [9], based on the core ontologies available to all components. The conversion to RDF can be implemented in client APIs if necessary. Thus, in our scenario, the service collects and stores information about past events (i.e., job executions and resource usage), and provides mining and searching for these mentioned past events.

The collected status information is stored as RDF inside the HDR. The repository can then be used for extracting statistics- and knowledge-based information or predictions. In its simple form, the repository can answer SPARQL queries to provide historic information. Clients can use the extended SPARQL provided by Jena ARQ interface [10], and ask for various aggregations of data, such as average, maximum, minimum. Additionally, Pellet [11] or the Jena built-in reasoner can be applied on the data for simple inferencing. For example, the use of subclasses can help to flexibly select a range of resource types for querying.

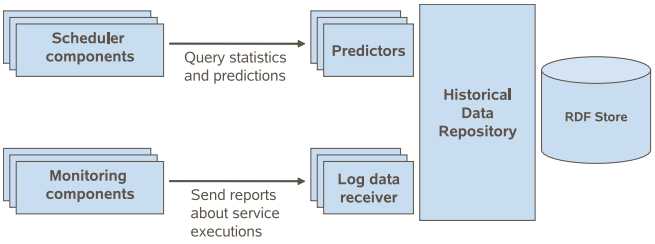


Figure 2. Historical Data Repository overview

For application-specific querying purposes a general plug-in mechanism has been developed. Predictors can be installed as plug-ins for HDR to answer specific questions. Within HDR, the RDF storage is coupled with the Weka data mining software [12]. In this way, a predictor can use both semantic querying and statistical methods. For example, a predictor can build a classification model on top of the results of a semantic query. The classifier can then be used to provide predictions based on the past.

In our specific scenario, the HDR is loaded with the description of the executed jobs, their resource usage and the resource downtimes. This data is used to train the classifier in order to provide estimations on the probabilities of delays in the schedule and problems with the job completion and estimations on the reliability of the used resources. As the statistical model is periodically updated, the provided values dynamically reflect the experiences of the recently finished executions.

D. Semantic Resource Allocation Process

The resource allocation for a particular job is decided between the JA and a set of RAs using the Contract Net Protocol (CNP) [13]. Figure 3 shows the message exchange between these agents for coordinating the job scheduling. When a JA has activated a goal for allocating resources to a job, it sends a query to the SMR to get the RAs whose resources match with the job requirements (1). Afterwards, the JA initiates a negotiation sending to the selected RAs a call for scheduling proposals (2). Each RA makes its own proposal and returns it to the JA (3). The JA evaluates all proposals, accepts the best for its interest and rejects the rest (4).

The RA proposals and the JA evaluations are done using an agent scheduler module. It is based on a rule engine evaluating a set of scheduling rules over semantic metadata bound to the GRO ontology (Section II-A). Despite of both agents using the same module, they behaves in a different way because they are loaded with different information and rules. The RA makes the scheduling proposals evaluating

 ISBN: 978-1-61208-106-9

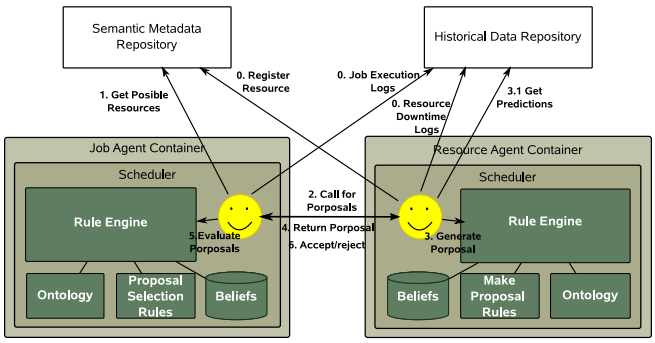


Figure 3. Distributed Semantic Scheduling

scheduling rules over the resource and assigned job descriptions, while the JA evaluates the scheduling proposal according to the customer rules and job description.

Once the RA has obtained a scheduling for a job in a resource, the RA consults the HDR to obtain predictions for this resource allocation (3.1). Resource allocation predictions are attached to the scheduling solution inferred by the scheduler to create a scheduling proposal, which will be evaluated by the JA. The following section gives more details about this process.

III. Job Prediction in Resource Allocation Process

In order to learn from past experiences, agents use the HDR as a service. In our HPC scenario, a separate HDR web service is applied for each Service Provider. This ensures that private internal data (such as log of execution details) remain within the boundaries of the provider, but also creates a merged knowledge base across various HPC clusters of the provider. A new plug-in was developed for the HDR, which is responsible for providing statistics and predictions for the scheduling agents. This predictor plug-in works on the basis of the common GRO explained in Section II-A containing the descriptions of hosts, software, host capabilities and QoS metrics.

When a new job is submitted to the system, it is described semantically indicating time constraints and the collection of resource requirements. During resource allocation process (described in Section II-D), the job is scheduled in the available resource and the results of this process are attached to the job description. During job execution, the JA monitors the execution introducing the relevant data in the job description (completion status, start time, end time and resource usage). Once the job has finished, the JA sends the job execution results to the HDR component using the HDR API client (Fig. 3, step 7). An example of this job execution report is shown in the following lines.

```
<rdf:Description rdf:about="gro:job-0">
  <gro:hasActionState rdf:resource="tech:done"/>
  <gro:hasResourceSet rdf:resource="gro:requirement_job-0"/>
  <rdf:type rdf:resource="gro:Execute_Task"/>
  <tech:hasDeadline>2010-06-14T19:41:07</tech:hasDeadline>
  <tech:expectedDuration>20</tech:expectedDuration>
  <gro:isExecutedAt rdf:resource="tech:host_1"/>
  <gro:incarnatedToProcess rdf:resource="gro:Incarnated_job-0"/>
</rdf:Description>

<rdf:Description rdf:about="gro:Incarnated_job-0">
  <rdf:type rdf:resource="gro:ProcessInstance"/>
  <tech:hasPlannedStart>2010-06-14T19:21:07</tech:hasPlannedStart>
  <tech:hasPlannedEnd>2010-06-14T19:41:07</tech:hasPlannedEnd>
  <gro:usesResource rdf:resource="tech:host_1"/>
  <gro:incarnatedFromTask rdf:resource="gro:job-0"/>
  <tech:hasActualStart>2010-06-14T19:21:07</tech:hasActualStart>
  <tech:hasActualEnd>2010-06-14T19:39:07</tech:hasActualEnd>
</rdf:Description>

<rdf:Description rdf:about="tech:usage_host_1_job-0">
  <tech:hasMeanMemoryUsage>25696.0</tech:hasMeanMemoryUsage>
  <tech:hasMeanCPUUsage>41720.0</tech:hasMeanCPUUsage>
  <tech:hasEnd>2010-06-14T19:39:07</tech:hasEnd>
  <tech:hasStart>2010-06-14T19:21:07</tech:hasStart>
  <tech:hasResource rdf:resource="tech:host_1"/>
  <tech:hasJob rdf:resource="gro:job-0"/>
  <rdf:type rdf:resource="tech:Usage"/>
</rdf:Description>
```

On the other hand, RAs report to the HDR about resource downtimes (Fig. 3, step 8). Job execution and resource downtime reports build up an extended log of actions inside the SP, which will be used as a knowledge base for generating the job predictions.

```
<rdf:Description rdf:about="tech:Downtime_host_1_1">
  <tech:hasEnd>2010-06-13T19:52:05</tech:hasEnd>
  <tech:isCausedBy rdf:resource="tech:powerOut"/>
  <tech:hasStart>2010-06-13T19:44:05</tech:hasStart>
  <tech:hasResource rdf:resource="tech:host_1"/>
  <rdf:type rdf:resource="tech:DownTime"/>
</rdf:Description>
```

Based on these data, a statistical classifier is trained, which can provide estimations on the probability of delays in the schedule, probability of problems with the job completion and on the reliability of the used resources. The relevant data for creating the statistical model is obtained from the HDR RDF store by means of SPARQL queries. For instance, we get the number of resource failures for the different types of resources and jobs from making the predictions of resource reliability and the job failure probability. The planned start and end times can be compared with the real start and end time for calculating delays on the scheduling and job duration. The statistical model is periodically updated with the historical data of recent execution. In this way, predictions dynamically reflect the experiences of past executions in the defined time window. The mechanism demonstrates the coupling of semantic data processing with data mining, as a promising novel combination.

Predictions are queried by the RA during the resource allocation process in order to make its scheduling proposal. The HDR provides the probability of delays in the job schedule (assigned host and time slot) inferred by the RA, the probability of problems during execution based on past executions with similar descriptions on similar hosts, and the reliability of the assigned host. The RA introduces these job predictions in the scheduling proposals and sends them to the

 ISBN: 978-1-61208-106-9

JA in order to be analyzed according to customer rules. In this case, the JA calculates reliability cost for the proposed scheduling based on a pondered mean of the predictions provided by the HDR. Once the job reliability cost for each proposal is analyzed, the JA will select the most reliable host, which fullfills the job constraints.

IV. Evaluation

In our experiments, we used the data from HPC resources of Barcelona Supercomputing Center (BSC). The test environment contained 8 hosts running 2 types of software. A HDR plug-in was implemented for the specific prediction tasks of the experiments. The plug-in at start-up extracts the necessary data using a SPARQL query:

```
SELECT ?task ?plannedStart ?plannedEnd ?start ?end
?resource ?status ?mem ?cpu WHERE{
  ?proc rdf:type gro:ProcessInstance .
  ?proc gro:usesResource ?resource .
  ?proc gro:incarnatedFromTask ?task .
  ?task gro:hasActionState ?status .
  ?proc tech:hasActualStart ?start .
  ?proc tech:hasActualEnd ?end .
  ?proc tech:hasPlannedStart ?plannedStart .
  ?proc tech:hasPlannedEnd ?plannedEnd .
  ?usage tech:hasJob ?task;
  ?usage tech:hasMeanMemoryUsage ?mem;
  ?usage tech:hasMeanCPUUsage ?cpu .
  FILTER (?start > "2010-01-01T00:00:00")
}
```

The data selected for this case includes the scheduled start and end for the job, the actual start and end times, the host where the job was run, the status of job completion and the mean memory and CPU usage of the job. Then, the necessary Weka data structure is built, and the classifiers are configured and run. Further data arriving during the predictor is in use can be added incrementally to the plug-in.

The expected duration to complete the job was calculated using various classifiers built into Weka. The best results were achieved by the M5P regression tree and the KStar instance-based classifier. The mean absolute error for the predicted value was 43 and 74 seconds respectively, where the actual value range was between 240 and 2400 seconds (i.e. the mean error is 3% of the mean predicted value).

Similarly, the status field can be used to predict the probability of failure. In this respect, Bayes-style, decision tree and decision rules algorithms were equally good at classifying the job completion status correctly for 85% of the job executions.

Furthermore, agents could assess the reliability of hosts by asking their availability metric from the plug-in. The availability is calculated based on the list of downtimes for the host. This function is supported directly by SPARQL queries, without using prediction mechanisms.

The time to load the predictor in the experiments can be broken up to the time needed for fetching the necessary data from the semantic log store (SPARQL query), and the time for building the prediction model with Weka. The first action required 1-2 seconds of time, while the second action could be accomplished in about 90 ms for 500 rows of data up to 141 ms for 5000 rows of data. As it was expected, the prediction time was independent of the number of data rows, yielding the result in approximately 15 ms.

We performed further testing of the prediction algorithms with log data available from two public archives: the Parallel Workloads Archive [14], and the Grid Workloads Archive [15]. The mean absolute error of the predictions in this case could be forced below 7% of the mean of the predicted value by careful preprocessing of the data. The comparison with our own data revealed that more details about job execution can yield better predictions in general. In contrast to regression used in cited related work (Section V), we preferred the instance-based learning algorithms, which relate similar job executions with each other for prediction purposes, and thus provided better predictions on job properties.

V. Related Work

Foster et al raised the benefit of integrating the results from agents and grids research areas in [16]. Agents could improve the autonomy, flexibility and scalabity of current grid systems. Regarding the area of resource allocation and job scheduling, the multi-agent system researchers have been already focusing on this problem. Several solutions have been proposed, such as the ones based on market-control where each agent tries to maximize its benefit function and the market controls them, the social wellfare where the multi-agent system tries to maximize a collective benefit and other proposals like game and decision theory. In the market-based solutions, we would like to highlight proposals like Challenger [17], Tycoon [18], other studies more focused on grids such as TRACE [19] or ARAM [20] and also projects such as CatNets [21] or SORMA [22]. The work on wellfare engineering and game theory for multiagents resource allocation has been compiled in [23] and [24] respectively.

All of these solutions implement specific allocation policies for solving a problem with their advantatges and drawbacks, but they are static and cannot be changed easily. In our paper we do not try to offer a new solution for the allocation algorithms, but we have introduced the multi-agent solution in the Semantic Resource Allocation process leaving users the availability of extending or changing the policies. In our system, customers and providers can describe the scheduling rules that are the most convenient for their interests. Those policies will be combined during the negotiation, trying to get a solution which satisfies all the policies.

In this case we have also introduced the capability of taking into account predictions based on semantic historical data about how similar jobs have been executed on the different resources. It allows the user taking into account in their policies the information provided by themselves as

 ISBN: 978-1-61208-106-9

well as information about how the job execution is predicted by the system.

Predictions have been used also in other systems. The work in [25] describes an approach for predicting SLA values during the execution of WS-BPEL processes. At given points in the process, the collected QoS data are fed into a prediction engine, which provides predictions for numerical SLO values using neural networks. Predictions are presented on a graphical user interface and open facilities for manual interaction in the executing process. In [26], the authors present an architecture for event-based collection of historical data, and performing prediction on QoS data in a SOA environment. This approach does not use semantics and its focus on QoS is different than current paper.

VI. Conclusion

The paper emphasizes on the importance of using historical data by service and cloud providers. We present a generic approach and a re-usable solution for the collection and exploitation of historical log data produced by services. The heterogeneity of log data arriving from various resources calls for a semantic data representation, which can facilitate the unification of these data and a query mechanism supported by inference. Our approach demonstrates the coupling of semantic data processing with data mining as a promising novel combination.

Acknowledgment

This work is supported by the Ministry of Science and Technology of Spain and the European Union under contract TIN2007-60625 (FEDER funds), Generalitat de Catalunya under contract 2009-SGR-980 and the European Commission with FP6-IST project 34556 (BREIN) and FP7-ICT project 215483 (S-CUBE).

References

[1] M. Papazoglou and D. Georgakopolous, "Service-oriented computing," *Communications of the ACM*, vol. 46, no. 10, p. 25, 2003.

[2] Y.Chevaleyre, et al, "Issues in Multiagent Resource Allocation," *Informatica*, vol. 30, pp. 3–31, 2006.

[3] "Java Agent Development Framework," http://jade.tilab.com[1].

[4] J. Brooke, D. Fellows, K. Garwood, and C. Goble, "Semantic matching of grid resource descriptions," in *Grid Computing*. Springer, 2004, p. 240.

[5] BREIN Consortium, "Final Report on the BREIN Core Ontologies," BREIN Project, Public Deliverable D3.2.5, 2008.

[6] J. Ejarque, et al, "Exploiting semantics and virtualization for SLA-driven resource allocation in SP," *Concurrency and Computation: Practice and Experience*, vol. 22, no. 5, p. 541, 2010.

[7] A. Rao and M. Georgeff, "BDI agents: From theory to practice," in *The 1st Int. Conf. on Multi-agent Systems*, 1995, p. 312.

[8] "Jadex system," http://jadex.informatik.uni-hamburg.de[1].

[9] "Resource Description Framework," http://www.w3.org/RDF[1].

[10] "ARQ - A SPARQL Processor for Jena," http://jena.sourceforge.net/ARQ[1].

[11] "Pellet OWL reasoner," http://clarkparsia.com/pellet/[1].

[12] I. Witten and E. Frank, *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann Pub, 2005.

[13] R. Smith, "The contract net protocol: High-level communication and control in a distributed problem solver," *IEEE Trans. on Computers*, vol. 100, no. 29, pp. 1104–1113, 1980.

[14] "Parallel workload archive," http://www.cs.huji.ac.il/labs/parallel/workload[1].

[15] "Grid workload archive," http://gwa.ewi.tudelft.nl[1].

[16] I. Foster, N. Jennings, and C. Kesselman, "Brain meets brawn: Why grid and agents need each other," in *The 3rd Int. Conf. on Autonomous Agents and Multiagent Systems*, 2004, p. 15.

[17] A. Chavez, A. Moukas, and P. Maes, "Challenger: A multiagent system for distributed resource allocation," in *The 1st Int. Conf. on Autonomous Agents*, 1997, p. 331.

[18] K. Lai, L. Rasmusson, E. Adar, L. Zhang, and B. Huberman, "Tycoon: An implementation of a distributed, market-based resource allocation system," *Multiagent and Grid Systems*, vol. 1, no. 3, pp. 169–182, 2005.

[19] S. Fatima and M. Wooldridge, "Adaptive task resources allocation in multi-agent systems," in *The 5th Int. Conf. on Autonomous Agents*, 2001, p. 544.

[20] S. Manvi, M. Birje, and B. Prasad, "An agent-based resource allocation model for computational grids," *Multiagent and Grid Systems*, vol. 1, no. 1, p. 17, 2005.

[21] "CatNets Project," http://www.catnets.uni-bayreuth.de[1].

[22] "Sorma Project," http://www.sorma-project.eu[1].

[23] Y. Chevaleyre, U. Endriss, S. Estivie, and N. Maudet, "Welfare engineering in practice: On the variety of multiagent resource allocation problems," *Engineering Societies in the Agents World V*, p. 335, 2005.

[24] S. Parsons and M. Wooldridge, "Game theory and decision theory in multi-agent systems," *Autonomous Agents and Multi-Agent Systems*, vol. 5, no. 3, p. 243, 2002.

[25] P. Leitner, et al, "Runtime Prediction of SLA Violations for Composite Services," in *3rd Workshop on Non-Functional Properties and SLA Management in SOC*, 2009.

[26] L. Zeng, C. Lingenfelder, H. Lei, and H. Chang, "Event-driven QoS prediction," in *6th Int.Conf on SOC*, 2008, p. 147.

[1]Last access to links in the references is August 18th, 2010

 ISBN: 978-1-61208-106-9

Storage QoS Aspects in Distributed Virtualized Environments

Darin Nikolow*, Renata Słota*, Jacek Kitowski*†
*Department of Computer Science, AGH-UST,
al. Mickiewicza 30, 30-059, Kraków, Poland
†Academic Computer Center CYFRONET-AGH,
ul. Nawojki 11, 30-950 Kraków, Poland
email: {darin,rena,kito}@agh.edu.pl
phone: (+48 12) 617 3964, fax: (+48 12) 338 054

***Abstract*—Storage performance of a single virtual machine in a cloud computing environment may be affected by other machines using the same physical storage. At the same time, user requirements concerning quality of service continue to increase, which brings new challenges for virtualized environments. In this paper we present the results of our research concerning data storage QoS. We discuss the quality aspects of storage services and propose a method for ensuring storage QoS by using monitoring and performance prediction based on heuristics. We also propose several heuristic policies, which can be implemented in a data transfer scheduler. Finally, we discuss test results obtained in a virtualized environment.**

Keywords-Quality of Service; Service Level Agreement; Storage Cloud; virtualization.

I. INTRODUCTION

Virtualization technologies are becoming extremely popular. One of the reasons behind this evolution is the increased flexibility in creating new computing environments for research, development or production. Another reason is that virtualization supports green computing by allowing for more efficient use of physical resources – this, in turn, reduces the need for electrical energy.

Since virtual machines share physical resources, the performance of a single machine may be affected by other machines running on the same host. It should also be noted that user requirements concerning QoS (Quality of Service) continue to increase, bringing new challenges for virtualized environments.

In the case of scientific applications (such as simulations, experiments in high energy physics or out-of-core computations) where the application needs to access huge amounts of data, as well as in the scope of backup and archiving services (for example storage clouds), proper management of data (or files), supporting QoS with efficient storage performance utilization, remains essential. Taking into account the dynamic nature of virtualized environments and the diversity/heterogeneity of storage systems (such as Hierarchical Storage Management – HSM and disk arrays), monitoring is required to cope with the problem of delivering data with appropriate QoS. Relevant policies for controlling and distributing data transfer requests are necessary for achieving QoS.

This paper presents the results of our research concerning storage QoS delivery. We discuss quality aspects of data storage services and propose a method of achieving storage QoS by using monitoring and performance prediction based on heuristics. We also present several heuristic policies, which can be implemented in a data transfer scheduler and discuss test results obtained in a virtualized environment.

The presented research forms part of a more general project called OntoStor [1], which deals with data access optimization. As a result of the project, an object-based mass storage model for HSM systems and disk arrays, called CMSSM (Common Mass Storage System Model) [2] has been proposed. A CIM (Common Information Model) version of CMSSM is also available. Implementation of monitoring sensors basing on the proposed model allows us to create data access estimators required to ensure storage QoS via proper management of storage resources.

This paper is organized as follows. The following section presents the state of the art. Subsequently, methods for achieving storage QoS are described. Section 4 describes test results, while the final section summarizes our research contributions.

II. STATE OF THE ART

As virtualization technologies are becoming more mature, a Cloud computing paradigm has evolved [3]. In terms of data storage, we can distinguish two types of clouds: the Storage Cloud (providing services for file-based or block-based access to data) and the Data Cloud (providing services for database-oriented access).

Examples of storage clouds include the Hadoop distributed file system [4], Amazon's S3 storage cloud [5], Google file system [6], Sector [7], and Chelonia [8].

Some studies on storage clouds involve high-performance data access, which is correlated with our research. Storage clouds, as well as data clouds, can be used as a data access layer for cloud computing. In [7], an implementation of the Sector data cloud is presented. Sector is a high-performance data cloud, which can operate on geographically-distributed data shared between data centers.

 ISBN: 978-1-61208-106-9

A study concerning data access prediction has been presented in [9]. The authors define five elements of the I/O path essential for end-to-end storage performance prediction, from device I/O scheduling through the cache and network layers, to client-side buffering. The aim of this research is to reduce the complexity of prediction. For example, in [10] and [11], the SAN (Storage Area Network) is addressed. The former paper [10] shows how performance management can be approached in IBM TotalStorage products and how it has evolved. In the latter one [11], SAN configuration middleware is introduced. The middleware aids the management of heterogeneous SAN deployments to meet customer SLA Service Level Agreement).

There are some papers concerning storage QoS. In [12], an I/O scheduling algorithm for managing the performance of an underlying storage device is presented. Scheduling is performed in such a way as to enable multiple users (or applications) to obtain the requested transfer-rate QoS.

In [13], a two-level framework for I/O scheduling, using monitoring and queuing theories, is presented. The proposed scheduler maintains QoS in terms of latency and transfer rates. Monitoring of underlying storage devices is used to provide feedback to the I/O scheduling algorithm.

In [14], a method for optimizing the workload distribution in storage area networks in order to meet SLA is presented. The method is static in the sense that the result of optimization is a concrete SAN configuration, not subject to frequent changes.

In [15], a method for storage performance insulation of services sharing common storage servers is proposed. The system automatically configures prefetching, write-back and cache partitioning to achieve storage performance for a given service, which is independent from the I/O activity of other services sharing the same storage.

In [16], the Chameleon framework capable of providing predictable performance for multiple clients sharing a common storage infrastructure is introduced. It bases on a black-box approach for capturing the behavior of the underlying storage system.

The presented studies show that the problem of achieving QoS in distributed storage environments is not a trivial task. Given the changing state of the environment, relevant monitoring is required. In our previous studies we have proposed a Common Mass Storage System Model for HSM systems [17], disk arrays and local disks. The aim of this paper is to present a unified monitoring layer, which can be used by estimation services for prediction of data access to mass storage systems [18][19]. The proposed model has been used in the NDS (National Data Storage) [18] and Platon [20] projects as well as in the PL-Grid project [21].

III. Problem solution description

We assume that the data needed by applications running in the Cloud is stored in a distributed storage system where it can be replicated for performance or availability reasons. The nodes of the storage system can also be located in the Cloud as VMs (Virtual Machines), sharing common resources. Applications may have various requirements concerning QoS. Some applications, such as HEP tools, which process data in the context of ongoing experiment require write QoS in order to prevent buffer overflows and data loss. In contrast, read QoS is needed by applications, which deliver on-demand multimedia content and by interactive applications dealing with large data sets, for example medical visualizations. In the case of write QoS, a decision on which storage node to use has to be made, while in the case of read QoS, an existing replica needs to be selected. Our vision of QoS-aware distributed storage environment is presented below.

A. QoS-aware distributed storage environment - a vision

The concept of a QoS-aware distributed storage environment is shown in Figure 1. The mass storage system layer may include divergent storage systems (HSM systems, disk arrays, local hard disks). They can be either complete systems with appropriate management software (HSM, disk arrays) or single storage devices such as local disks. The sensor layer provides a unified interface for monitoring of storage performance parameters, facilitating the development of storage-independent upper layer services. The estimation layer contains estimation services, which can be used to predict the performance of a storage node at a given moment. The estimation services use monitoring data provided by the sensor layer and can apply various algorithms (more or less sophisticated) to predict the performance of a given storage node. The data management layer contains services needed for efficient access to data. Examples of such services include the meta-catalog, the replica manager and the QoS scheduler.

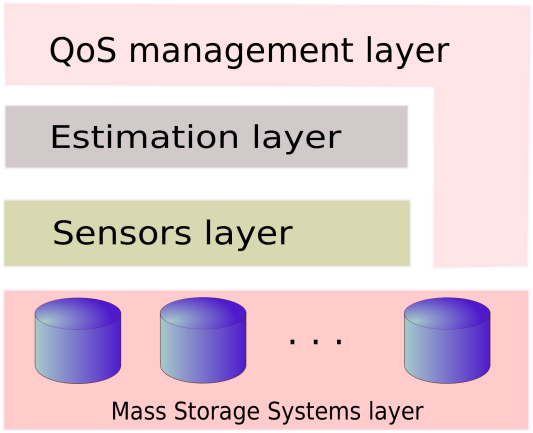


Figure 1. QoS-aware distributed Storage Environment - a vision

The problem that we address in this paper concerns efficient storage resource allocation. In order to effectively

 ISBN: 978-1-61208-106-9

use the available storage resources in terms of capacity and bandwidth, the storage system has to make decisions about scheduling and allocating storage nodes with QoS in mind. Storage-related QoS requirements include transfer rate and latency time. Several types of transfer rate can be specified in this context: current, average, moving average, etc. The QoS requirements used in our tests involve average transfer rates for a given transfer. For each data transfer request, a storage node is selected from a prepared list. The list only contains storage nodes suitable for the current request. In the case of a write request, it can include all nodes with the required level of availability or storage cost. In the case of read requests, it is limited to nodes containing replicas of the requested file. If the storage system is unable to fulfill the QoS for a given request, the request is postponed.

The proposed method of resource allocation with QoS uses bandwidth reservation along with storage performance monitoring. Such monitoring is necessary since the storage resources can be shared among VMs (for example, high-capacity disk arrays in a storage area network) and their performance can not be predicted based on reservations only. In addition, if an application does not fully exhaust its requested (or reserved) bandwidth, a wastage of storage performance may occur. Using proper monitoring we can detect such wastage and, conditionally, allocate bandwidth to other ongoing transfers. The proposed heuristic-based policies are presented below. They can be implemented in a QoS scheduler and shared by the management and estimation layers (see Figure 1).

B. Resource allocation policies with storage QoS delivery

The proposed monitoring-based heuristic policies designated SM-R1, SM-R2, SM-R3 are described below. Their usefulness has been tested (see next section) and compared with policies, which do not employ monitoring, i.e., RR (Round Robin) and StaticQoS.

The following monitoring-based heuristic policies have been considered:

- SM-R1 - scheduling with monitoring using the following rule to check if a node can meet the QoS requirements of a given transfer:

$$max - res\_ban \times k_1 > rb, \tag{1}$$

  where max is the peak transfer rate for the requested operation (read or write) concerning the given disk storage resource, res_ban is the amount of reserved bandwidth, k_1 is the overhead coefficient and rb is the requested bandwidth,
- SM-R2 - for this policy the rule is:

$$max - ma_{10} \times k_2 > rb, \tag{2}$$

  where ma_{10} is the moving average covering the last 10 transfer rate measurements for the given storage resource while k_2 is the overhead coefficient,
- SM-R3 - for this policy the checking rule is:

$$\frac{ma_{10}}{np+1} > rb \times k_3, \tag{3}$$

  where np is the number of scheduled active data transfers and k_3 is the overhead coefficient.

The monitoring-independent resource allocation policies are as follows:

- RR - round robin. Here, the next available storage node is selected for the current transfer.
- StaticQoS - the number of allowed concurrent transfers for the given node is limited and static. The limit λ is calculated using the following formula:

$$\lambda = \frac{BSN_{max}}{rb \times k_4}, \tag{4}$$

  where BSN_{max} is the peak bandwidth for a storage node and k_4 is the concurrent transfer overhead coefficient.

IV. Storage QoS test results

This section presents our test environment and the results of tests involving the previously-described policies. Tests have been conducted under idle and loaded conditions.

A. Test environment

Our testing environment consists of three storage nodes, each of which is a virtual machine with dedicated storage. All VMs are hosted by the same physical machine, thus sharing its hardware resources. The physical machine is part of the Cloud environment. This test environment allows us to study the manageability of storage QoS given the sharing of resources due to virtualization and the heterogeneity of storage systems. We should notice that hosting too many storage node VMs on a single physical machine may cause the I/O bus of the physical host to become a bottleneck, thereby wasting the performance capabilities of the attached storage systems.

The testing environment is presented in Figure 2. The first node is a client of the Lustre cluster filesystem, the second has a disk array volume mounted and the last one uses a local logical volume. The measured bandwidth of the attached storage is as follows: 20 MB/s for the Lustre FS, 300 MB/s for the disk array and 60 MB/s for the local disk. This gives us the peak total throughput of 380 MB/s. The low bandwidth of Lustre is due to the network link of OST (Object Storage Target). On every node a monitoring daemon has been started in order to gather storage performance statistics (peak, current and moving-average transfer rates).

The testing procedure was as follows:

1) select the most appropriate storage node for a transfer according to the tested policy (if there is no node capable of satisfying the QoS requirement then wait until there is one). The QoS requirement for all transfers has been set at 10 MB/s,

 ISBN: 978-1-61208-106-9

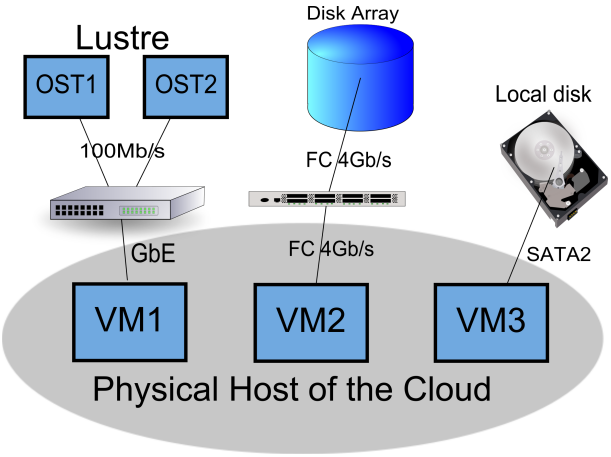


Figure 2. Test environment for storage QoS.

2) fork a write data transfer process for the selected node,
3) sleep 1 second (a configurable parameter),
4) repeat previous steps until the defined number of iterations is reached. For the presented results the number of iterations is set at 100. The number of concurrent transfers depends on the tested policy, e.g., the RR policy can potentially launch many more transfers then other policies since no condition is checked.

B. Results for idle system

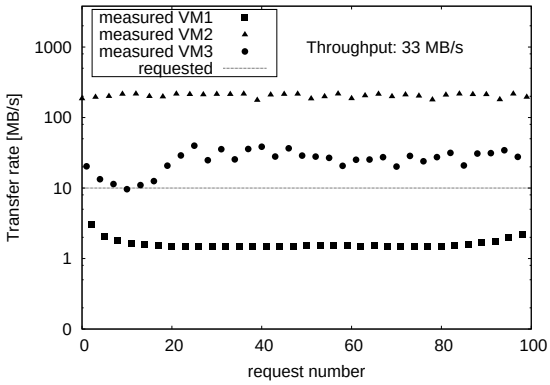
The test results are presented in Figure 3 and Figure 4.

We can see that in the case of the RR policy (see Figure 3(a)) most transfers did not receive the required bandwidth. In this case, we are dealing with very inefficient use of storage resources. Since all three SNs have divergent storage devices attached, we see three separate lines, which depict the performance of each device. The throughput is very low and limited by the device with the worst performance. In addition, sending more concurrent transfers further degrades the overall performance.

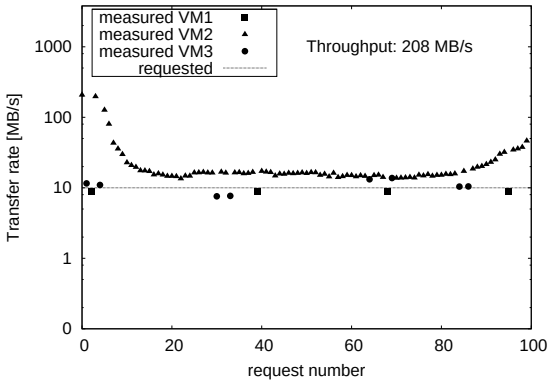
In the case of a static QoS policy (see Figure 3(b)) we can see that only several transfers are below the required bandwidth (though their performance remains quite close to expectations). Total throughput is much better than in the RR case. This simple policy may already be deemed satisfactory, but it might be possible to achieve even higher throughput.

In the case of the SM-R1 policy (see Figure 4(a)), in which monitoring is used along with resource reservation, we have high total throughput but more requests fall below the line compared to static QoS. This is a consequence of the fact that with concurrent transfers, the throughput of storage devices is usually lower than given a single transfer. The number of concurrent transfers for a given storage device is not monitored and so its influence is not predicted.

The SM-R2 policy (see Figure 4(b)) yields very low performance because the rule remains true even when many



(a) Round robin



(b) Static QoS

Figure 3. Test result for policies with no monitoring

concurrent transfers are ongoing. The average throughput ma_{10} (see Subsection III-B) is highly degraded and cannot approach max. This degradation is caused by the fact that data transfer processes need to compete for CPU time instead of waiting for I/O access.

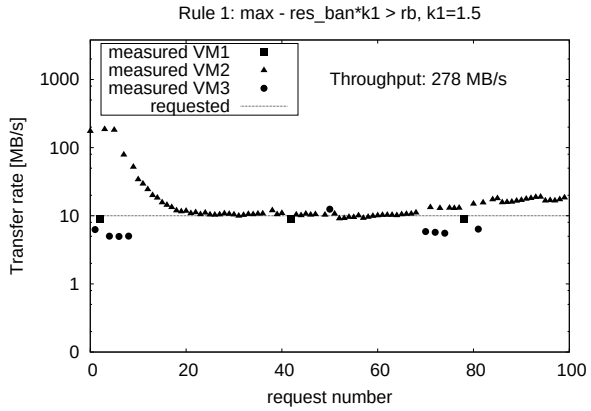
In the final case, SM-R3 (see Figure 4(c)), we have obtained the best results, but it should be mentioned that the rule proves false if there are no ongoing transfers - $ma_{10} = 0$. That is why an additional condition was introduced: if no transfers are present then start one.

C. Results for loaded conditions

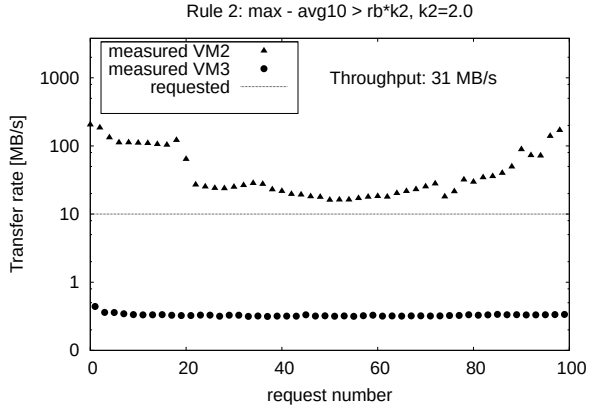
The same tests have been run in an environment where the disk was subjected to additional, external load.

The result for static QoS and SM-R3 are presented in Figure 5. These policies have been selected since they give the best results for idle storage. As can be seen, the extra load has a heavy impact on the static-policy QoS while in the case of SM-R3 it does not significantly increase the number of transfers, which fall bellow the 10MB/s line (the requested bandwidth). It should be noticed that even if the transfer rate is below 10MB/s it is still quite close to the required value. The total throughput for the SM-R3 policy

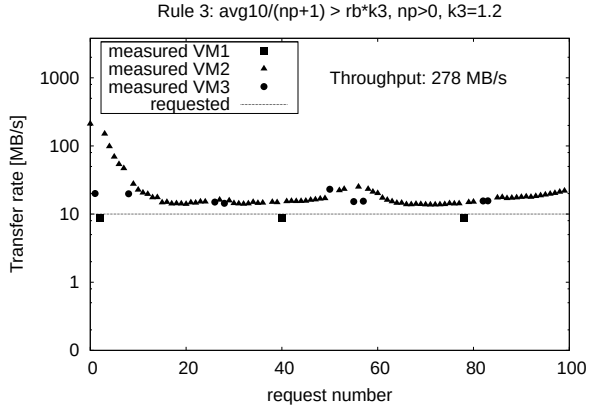
 ISBN: 978-1-61208-106-9



(a) SM-R1

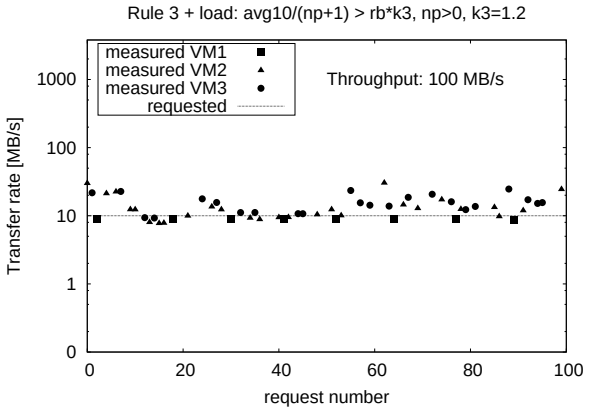


(b) SM-R2

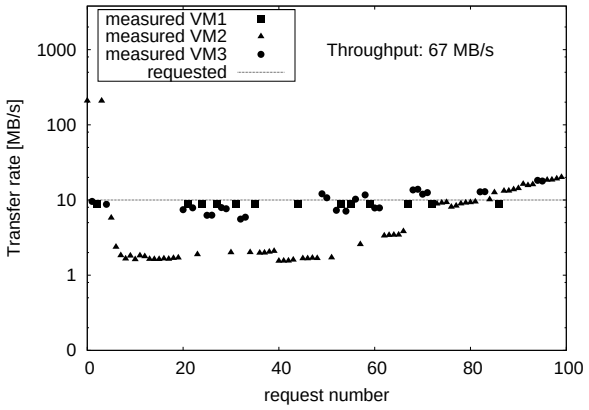


(c) SM-R3

Figure 4. Test result for policies using monitoring



(a) SM-R3 with external load



(b) Static QoS with external load

Figure 5. Test result for selected policies with external storage load

is also much better.

V. Conclusion and Future Work

This paper presented the results of our research concerning storage QoS delivery. A method of achieving storage QoS by using monitoring and performance prediction based on heuristics is proposed and several heuristic policies are defined. Test results for a virtualized environment show that scheduling data transfers with the aid of monitoring policies can yield better results than without monitoring. Policies with static resource reservation do not perform well in a virtualized environment with shared resources, similar to a storage cloud. In such cases, storage performance monitoring and prediction is a must, especially when managing storage resources with QoS (or SLA). The problem is not trivial given the heterogeneity of storage systems and the lack of relevant performance parameters in modern information models like CIM and GLUE (Grid Laboratory for a Uniform Environment). While methods of performance monitoring and estimation, as well as their impact on storage systems have been discussed in our previous work, this paper presents a vision of how they can be applied for the purpose of delivering storage QoS.

Acknowledgments

This research is supported by Polish MNiSW grant no. N N516 405535. The AGH-UST grant no. 11.11.120.865 is also acknowledged.

REFERENCES

[1] OntoStor project. [Online]. Available: http://www.icsr.agh.edu.pl/ontostor/17.08.2010

[2] "Common Mass Storage System Model - project Ontostor internal report," 2009. [Online]. Available: http://www.icsr.agh.edu.pl/trac/ontostor/browser/CMSSM/raport.odt17.08.2010

[3] K. Kant, "Data center evolution," *Comput. Netw.*, vol. 53, no. 17, pp. 2939–2965, 2009.

[4] Welcome to Apache Hadoop! [Online]. Available: http://hadoop.apache.org/17.08.2010

[5] Amazon Simple Storage Service (Amazon S3). [Online]. Available: http://aws.amazon.com/s3/17.08.2010

[6] S. Ghemawat, H. Gobioff, and S.-T. Leung, "The Google file system," *SIGOPS Oper. Syst. Rev.*, vol. 37, no. 5, pp. 29–43, 2003.

[7] Y. Gu and R. L. Grossman, "Sector and Sphere: the design and implementation of a high-performance data cloud," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 367, no. 1897, pp. 2429–2445, 2009. [Online]. Available: http://rsta.royalsocietypublishing.org/content/367/1897/2429.abstract

[8] J. K. Nilsen, S. Toor, Z. Nagy, B. Mohn, and A. L. Read, "Performance and Stability of the Chelonia Storage Cloud," *CoRR*, vol. abs/1002.0712, 2010.

[9] D. O. Bigelow, S. Iyer, T. Kaldewey, R. C. Pineiro, A. Povzner, S. A. Brandt, R. A. Golding, T. M. Wong, and C. Maltzahn, "End-to-end performance management for scalable distributed storage," in *PDSW*, G. A. Gibson, Ed. ACM Press, 2007, pp. 30–34.

[10] S. Gopisetty, S. Agarwala, E. Butler, D. Jadav, S. Jaquet, M. Korupolu, R. Routray, P. Sarkar, A. Singh, M. Sivan-Zimet, C.-H. Tan, S. Uttamchandani, D. Merbach, S. Padbidri, A. Dieberger, E. M. Haber, E. Kandogan, C. A. Kieliszewski, D. Agrawal, M. Devarakonda, K.-W. Lee, K. Magoutis, D. C. Verma, and N. G. Vogl, "Evolution of storage management: transforming raw data into information," *IBM J. Res. Dev.*, vol. 52, no. 4, pp. 341–352, 2008.

[11] D. M. Eyers, R. Routray, R. Zhang, D. Willcocks, and P. Pietzuch, "Towards a middleware for configuring large-scale storage infrastructures," in *MGC '09: Proceedings of the 7th International Workshop on Middleware for Grids, Clouds and e-Science*. New York, NY, USA: ACM, 2009, pp. 1–6.

[12] T. M. Wong, R. A. Golding, C. Lin, and R. A. Becker-szendy, "Zygaria: storage performance as a managed resource," in *In IEEE Real Time and Embedded Technology and Applications Symposium (RTAS 06*, 2006, pp. 125–134.

[13] J. Zhang, A. Sivasubramaniam, Q. Wang, A. Riska, and E. Riedel, "Storage performance virtualization via throughput and latency control," *Trans. Storage*, vol. 2, no. 3, pp. 283–308, 2006.

[14] E. Gencay, C. Sinz, and W. Kuchlin, "Towards SLA-based optimal workload distribution in SANs," in *Network Operations and Management Symposium, 2008. NOMS 2008. IEEE*, 7-11 2008, pp. 755 –758.

[15] M. Wachs, M. Abd-el-malek, E. Thereska, and G. R. Ganger, "Argon: Performance insulation for shared storage servers," in *In Proceedings of the 5th USENIX Conference on File and Storage Technologies. USENIX Association*. USENIX Association, 2007, pp. 61–76.

[16] S. Uttamchandani, L. Yin, G. A. Alvarez, J. Palmer, and G. Agha, "CHAMELEON: a self-evolving, fully-adaptive resource arbitrator for storage systems," in *ATEC '05: Proceedings of the annual conference on USENIX Annual Technical Conference*. Berkeley, CA, USA: USENIX Association, 2005, pp. 75–88.

[17] D. Nikolow, L. R. Slota, and J. Kitowski, "Grid Services for HSM Systems Monitoring," in *in: R. Wyrzykowski, J. Dongarra, K. Karczewski, and J. Wasniewski (Eds.), Proceedings of 7-th International Conference, PPAM 2007*. Springer, 2008, pp. 321–330.

[18] R. Słota, D. Nikolow, S. Polak, M. Kuta, M. Kapanowski, K. Skałkowski, M. Pogoda, and J. Kitowski, "Prediction and Load Balancing System for Distributed Storage," *Scalable Computing: Practice and Experience*, vol. 11, no. 2, pp. 121–130. [Online]. Available: http://www.scpe.org

[19] D. Nikolow, R. Slota, J. Marmuszewski, M. Pogoda, and J. Kitowski, "Disk Array Performance Estimation," in *Proceedings of Cracow Grid Workshop - CGW'09*. ACC-Cyfronet AGH, 2010, pp. 287–294.

[20] PLATON - Service Platform for e-Science. [Online]. Available: http://www.platon.pionier.net.pl/online/?lang=en17.08.2010

[21] PL-Grid Polish Grid Initiative. [Online]. Available: http://www.plgrid.pl/en17.08.2010

 ISBN: 978-1-61208-106-9

Self-scaling the Cloud to meet Service Level Agreements

An open-source middleware framework solution

Antonin Chazalet, Frédéric Dang Tran,
and Marina Deslaugiers
France Telecom - Orange Labs,
28 chemin du vieux chêne,
38240 Meylan, FRANCE,
{antonin.chazalet, frederic.dangtran,
marina.deslaugiers}@orange-ftgroup.com

François Exertier, and Julien Legrand,
Bull,
1, rue de Provence,
B.P. 208
38432 Echirolles Cedex, FRANCE,
{francois.exertier, julien.legrand}@bull.net

***Abstract* - Cloud computing raises many issues about Virtualization and Service-Oriented Architecture (SOA). Topics to be addressed regarding services in Cloud computing environment include contractualization, monitoring, management, and autonomic management. Cloud computing, promotes a "pay-per-use" business model. This business model should enable to reduce costs but requires flexible services than can be adapted to load fluctuations. This work is conducted in the European CELTIC Servery cooperative research project, which deals about a telecommunication services marketplace platform. The Servery project focuses amongst others on the self-scaling capability of Telco services in a cloud environment. The self-scaling capability is achieved thanks to Service Level Agreement (SLA) monitoring and analysis (i.e., compliance checking), and to autonomic reconfiguration performed according to the analysis results. SLAs are defined for the services and for the cloud virtualized environment. In order to achieve this self-scaling capability, a specialized autonomic loop is proposed. Our proposal is well in line with the Monitor, Analyze, Plan, and Execute loop pattern defined by IBM. The proposed solution is based on the following open-source middleware: Service Level Checking, OW2 JASMINe Monitoring, OW2 JASMINe VMM. This paper presents this solution that has been implemented and validated in the context of the Servery project.**

Keywords* - *Cloud Computing; Autonomic; Self-Scaling; Service Level Agreement; Service Level Checking; Virtualization; Open-source.

I. INTRODUCTION

Today, almost all IT and Telecommunications industries are migrating to a Cloud computing approach. They expect that the Cloud computing model will optimize the usage of physical and software resources, improve flexibility and automate the management of services (i.e., Software as a Service, Platform as a Service, and Infrastructure as a Service). Cloud computing is also expected to enable data centers subcontracting from Cloud providers.

As a consequence, Cloud computing is seen as a way to reduce costs via the introduction and the use of "pay per use" contracts. It is also seen as a way to generate incomes for Cloud providers. Note that a Cloud provider can be:

- An infrastructure provider,
- A platform provider,
- And/or a software provider.

Cloud computing raises many issues. Many of them are related to Virtualization, and Service-oriented Architecture (its implementation and its deployment). These issues lie at both the hardware and/or software levels.

Nevertheless, Cloud computing also raises issues related to services contractualization, services monitoring, services management, and autonomic for the Cloud. In this paper, we address these last issues for telecommunication services offered to customers through a Cloud. These issues are critical: indeed economical concerns (i.e., the establishment and the use of the pay-per-use contracts) require the ability to contractualize services (via the use of service level agreements: SLA), to monitor and manage them, to check services contracts compliance, and to manage virtualized environments.

This paper is organized as follows. The next section provides background about Autonomic computing and Service Level Checking. Section 3 presents the related works. Section 4 outlines the autonomic approach we have followed. Section 5 focuses on the targeted Servery use case. Section 6 details our open-source solution. Section 7 describes the implementation. We present the validation and the results obtained in Servery in Section 8. Last section concludes this paper and gives directions for future works.

II. BACKGROUND

This section presents background about autonomic computing and service level checking.

A. Autonomic computing

Autonomic computing refers to computing systems (i.e., autonomic managers) that are able to manage themselves or others systems (i.e., managed resources) in accordance to management policies and objectives [1]. Thanks to automation, the complexity that human administrators are facing is moved into the autonomic managers. It allows administrators to concentrate on high-level management objectives definition and no more on the ways to achieve theses objectives. In [2], the authors define principles of autonomic computing thanks to a biological analogy with the

 ISBN: 978-1-61208-106-9

human nervous system: a human can achieve high level goals because its central nervous system allows him to avoid spending time on managing repetitive and vital background tasks such as regulating its blood pressure.

[2] specifies four main characteristics for describing systems self-management capabilities:

- Self-Configuration that aims to automate managed resources installation, and (re-)configuration.
- Self-Healing that purposes to discover, diagnose and act to prevent disruptions. Here, note that self-repair is a part of self-healing.
- Self-Protect that aims to anticipate, detect, identify and protect against threats.
- Self-Optimize that purposes to tune resources and balance workloads to maximize the use of information technology resources. Self-scaling is a subpart of self-optimization.

B. *Service Level Checking*

Generally speaking, Service Level Checking (SLC) involves a target service and a system in charge of collecting monitoring information and checking SLA compliance. More precisely, the target service offers probes, and its usage (or a derived usage) is contractualized with at least one SLA. SLA definitions are based on information that can be obtained through the services probes (directly or via calculation). The SLC system takes as input information regarding the target service as well as at least one SLA, and produces SLC results about the SLA compliance, the SLA violation, or errors that occurred during the checking or information collection steps. The target service can include software, platform and/or infrastructure, or can even be a Cloud itself (i.e., a set of software services, platform services and infrastructure services).

The SLC results can be used to inform a service administrator, to select a service provider at runtime, to launch an autonomic loop, and/or to break a contract.

III. RELATED WORKS

This related works section describes the solutions for autonomic computing proposed by equipment and IT vendors (i.e., IBM, Oracle, HP, Motorola, Cisco, Alcatel-Lucent ...). The focus is set on the use of SLA based service level checking as analyzing part (in autonomic MAPE loop) and the ability to manage virtualized environments (mandatory today in Cloud computing).

First, IBM uses policies managers as analyzers for the MAPE loop. IBM promotes the use of the Simplified Policy Language (SPL). SPL is based on Boolean algebra, arithmetic functions and collections operations. SPL also uses conditional expressions [3]. IBM Tivoli System Automation targets the reduction of the frequency and of the duration of service disruptions. It uses advanced policy-based automation to enable the high availability of applications and middleware running on a range of hardware platforms and operating systems. Note that these platforms and systems can be virtualized (or not). Tivoli's products family targets mainly availability and performance [4].

Second, Oracle provides the WebLogic Diagnostics Framework in order to detect SLA violations [5]. The Oracle Enterprise Manager 10g Grid Control can monitor services and report on service availability, performance, usage and service levels. Note that it doesn't manipulate SLA but a similar concept named Service Level Rule [6]. Oracle Enterprise Manager 11g Database Management is a solution to manage databases in 24x7. It self-tunes and self-manages databases operating w.r.t the performance, and it provides proactive management mechanisms (that involve service levels) in order to avoid downtime and/or performance degradation [7]. Oracle handles and manages virtualization through its Oracle VM Management Pack [8]. Oracle also leads research concerning PaaS and the Cloud, and provides a product called Oracle Fusion Middleware (OFM) [9]. OFM targets amongst others management automation, automated provisioning of servers, automate system adjustments as demand/requirements fluctuates. Note that unlike [1], Oracle specifies only three steps for the autonomic loop: Observe, Diagnose, and Resolve [10].

Third, autonomic architectures proposed by other equipment and IT vendors focus mainly on basic autonomic features in IT products [11]. It also shows that these remaining architectures don't use policies managers or SLA based service level checking as analyzing part, and don't manage virtualized environments.

The coming sections illustrate that our solution is well in line with the MAPE loop pattern. It uses a SLA based SLC as analyzing part and it manages virtualized environments. Moreover, unlike IBM and Oracle, it is an open-source solution: indeed, it only involves open-source middleware.

IV. APPROACH

The approach followed in this work is well in line with the Monitor, Analyze, Plan, and Execute loop pattern defined by IBM: the MAPE loop pattern (see Figure 1).

In [1], the authors defined that, similarly to a human administrator, the execution of a management task by an autonomic manager can be divided into four steps (that share knowledge):

- Monitor: The monitor function provides the mechanisms that collect, aggregate, filter and report details (such as metrics and topologies) collected from a managed resource.
- Analyze: The analyze function provides the mechanisms that correlate and model complex situations (with regard to the management policy). These mechanisms enable the autonomic manager to learn about the IT environment and help predict future situations.
- Plan: The plan function provides the mechanisms that construct the actions needed to achieve goals and objectives. The planning mechanism uses policy information to guide its work.
- Execute: The execute function provides the mechanisms that control the execution of a plan with considerations for dynamic updates.

 ISBN: 978-1-61208-106-9

These four parts work together to provide the control loop functionality.

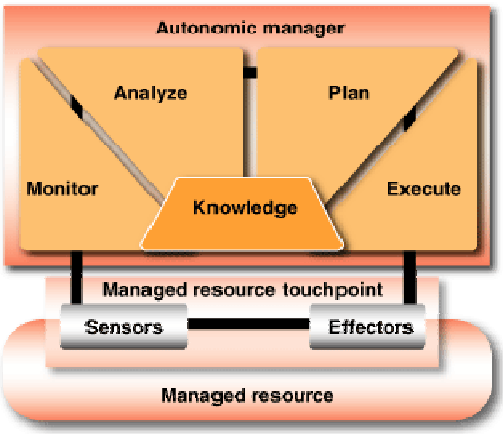


Figure 1. Autonomic loop (or MAPE/MAPE-K loop) [1].

V. THE SERVERY USE CASE

This section presents the Servery project description and the Servery self-scale-up use case.

A. *Servery research project description*

This sub-section presents the Servery research project context.

First, Servery (Service Platform for Innovative Communication Environment) is addressing the still unsolved problem of designing, developing and putting into operation efficient and innovative mobile service creation/deployment/execution platforms for networks beyond 3G [12]. One of the main goals of Servery is to propose a services marketplace platform where Telco services can be executed, and where end users can search, browse and access the executed services. Note that services published in the Servery marketplace platform can also be executed in others platforms belonging, e.g., to the telecommunication operators themselves.

The Figure 2 below shows an overview of Servery's context diagram, i.e., end users that are external actors of the system use Telco services provided by the Servery Marketplace Platform.

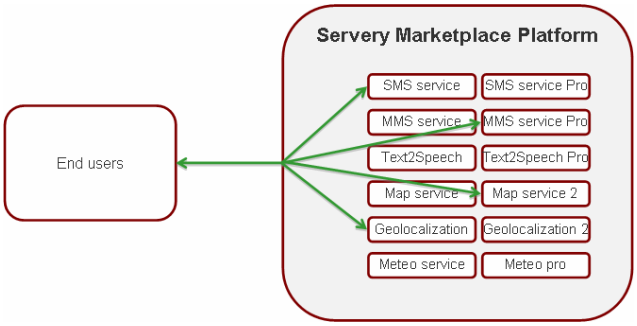


Figure 2. Servery's context diagram.

B. *Servery self-scale-up use case*

This sub-section presents the Servery self-scale-up use case. The goal of this use case is to maintain the overall QoS of the services executed in the Servery marketplace platform and of the marketplace platform itself while the user load grows up. QoS is directly (and indirectly) defined via SLAs. Here, the user load is represented by the number of end users (and consequently by the number of requests sent). The services targeted are Telco services, e.g., SMS services, e-mail services, etc.

VI. SOLUTION FOR SERVERY SELF-SCALING

As presented in the related works section, our proposition is well in line with the MAPE loop pattern defined in [1]. Our idea is to define SLAs between the administrators of the Servery marketplace platform and the marketplace platform itself. The whole MAPE loop proposed is based on these defined SLAs. It is named Servery marketplace management platform. Its monitoring and analyzing parts depend directly on the elements and metrics specified in the SLAs. As a reminder, the Servery marketplace platform is a Cloud. It means that three types of entities can be distinguished: the entities belonging to the software level, the platform entities and the infrastructure entities. SLAs defined can specify information related to these three types of entities.

More precisely, the analyzing part contains two distinct sub-parts: the SLC [13], and the JASMINe Monitoring (and its Drools module) [14]. The SLC is in charge of requesting the relevant probes and collecting the monitoring data. It is also in charge of checking the compliance of the defined SLAs with the collected monitoring data. It produces SLC results about the SLA compliance, the SLA violation, or errors occurred during the checking or information collection steps. JASMINe Monitoring takes these SLC notifications as input and checks their frequency over a configurable sliding time slot. This analysis over a sliding time slot is realized by a Drools module. Drools is a business logic integration platform which provides a unified and integrated platform for rules, workflow and event processing [15]. Using a sliding time slot analysis is interesting because it avoids to launch the planning and executing steps for non-significant/non-relevant events.

JASMINe Monitoring is also in charge of the planning part and leads the execution part. All the execution actions related to the virtual machines management is done via the mechanisms provided by JASMINe Virtual Machines Management (JASMINe VMM) [16].

The Servery marketplace platform (see Figure 3) was designed with a front-end element (i.e., an Apache HTTP Server) and at least one services execution environment (i.e., an OW2 JOnAS open-source Java EE 5 Application Server [17]). This design allows us to be able to scale-up the Servery marketplace platform and the Telco services deployed in it. In short, the Apache front-end acts as a load balancer. Note that the Apache front-end and all the JOnAS server(s) are run in virtual machines themselves run over the Xen hypervisor technology - an open source industry standard for virtualization [18].

 ISBN: 978-1-61208-106-9

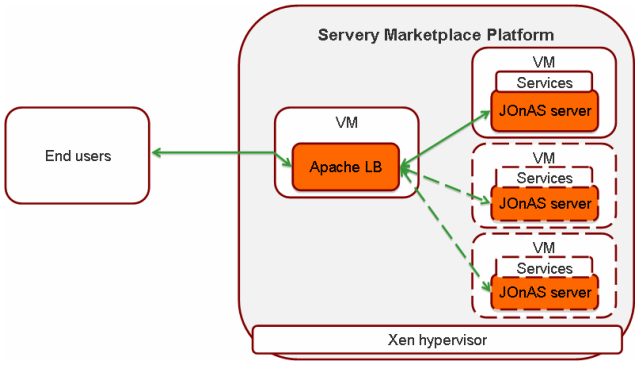


Figure 3. Servery marketplace platform architecture.

This marketplace platform design is interesting, because it enables to easily support the addition and/or removal of services execution environments. The only constraint of this design is the need to reconfigure the front-end element in order to take into account the addition and/or removal.

VII. IMPLEMENTATION

This section presents the implementation of the proposed solution. Our solution involves two high level modules: the Servery marketplace platform that is in charge of providing services to the end users, and the Servery marketplace management platform that ensures the scale-up autonomic property.

The Servery marketplace management platform involves four distinct modules:

- Service level checking is in charge of requesting the relevant probes and collecting the monitoring data from the Servery marketplace platform. It is also in charge of checking the compliance of the defined SLAs with the monitoring data collected. It produces SLC results that are sent to JASMINe monitoring. SLC is developed by France Telecom.
- JASMINe Monitoring is part of the OW2 JASMINe project. The OW2 JASMINe project aims to develop an administration tools suite dedicated to SOA middleware such as application servers (Apache, JOnAS, ...), MOM (JORAM, ...) BPM/BPEL/ESB solutions (Orchestra, Bonita, Petals, ...) in order to facilitate the system administration [19]. JASMINe Monitoring takes these SLC notifications as input and checks their frequency over a configurable sliding time slot. It is also in charge of the planning step and it leads the scale-up execution step. JASMINe Monitoring is developed by Bull.
- Cluster scaler is in charge of transmitting execution actions to JASMINe VMM. It is also in charge of the reconfiguration of the Apache Load Balancer in order to take into account the virtual machine just added. Cluster scaler is developed by Bull.
- JASMINe VMM is in charge of the management of the virtual machines created and executed over the Xen hypervisor. JASMINe VMM aims at offering a unified Java-friendly API and object model to manage virtualized servers and their associated hypervisor. In short, it provides a JMX hypervisor-agnostic façade/API in front of proprietary virtualization management protocols or APIs (such as the open-source Xen and KVM hypervisors, the VMware ESX hypervisor, the Citrix Xen Server hypervisor, and the Microsoft Hyper-V 2008 R2). JASMINe VMM is developed by France Telecom.

Note that the solution we propose is a fully open-source and Java based solution, and that all communications are done via the Java Management eXtension technology (JMX).

We now present the nominal steps executed when an autonomic scale-up is launched (see Figure 4). Here, the Servery Marketplace Platform initially contains two virtual machines (one containing the Apache LB, and one containing a JOnAS server and Telco services). End users request/interact with the (services of the) Servery marketplace platform is referred as step number 0. A nominal execution involves 6 steps (from 1 to 6):

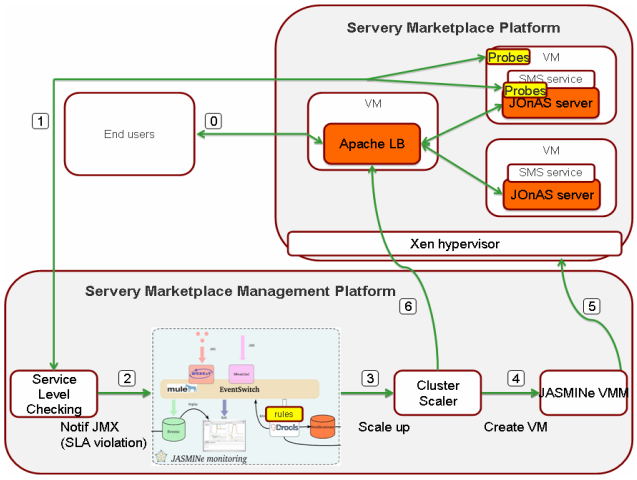


Figure 4. Overview of the proposed solution.

First, the objective of SLC is to check the compliance of the Servery Marketplace Platform (and its Telco services) with SLAs related: to the SaaS level (i.e., Telco services level), to the PaaS level (i.e., JOnAS server level), and to the IaaS level (i.e., virtual machines level). Consequently, SLC requests probes related to the Telco services, the JOnAS server and the virtual machine with regard to the contracts wanted. Amongst all the possible probes, we have chosen to focus and collect the following Telco services (SaaS) information:

- The number of requests processed during the last (configurable) time period
- The total processing time during the last period
- The average processing time during the last period

The JOnAS server information chosen was:

- The current server state (e.g., starting, running)
- The number of active HTTP sessions
- The number of services deployed/running in a server

 ISBN: 978-1-61208-106-9

The virtual machine information chosen was:

- The virtual machine CPU load
- The total memory (heap and no-heap)
- The used memory (heap and no-heap)

Second, SLC results are sent to JASMINe monitoring. JASMINe monitoring then checks the frequency of the SLC results corresponding to a violation. If the frequency of the violations is too high (e.g. more than five violations in a one minute sliding time slot), it means that a scale-up action is needed. So, JASMINe monitoring plans this scale-up action (thanks to information known about the marketplace platform) and executes it. Here, it means that JASMINe monitoring plans to introduce and configure another virtual machine containing a JOnAS server and the Telco services.

Third, the scale-up action is sent to the Cluster Scaler.

Fourth, Cluster Scaler commands the JASMINe VMM to create a new virtual machine (containing a JOnAS server and the Telco services).

Fifth, JASMINe VMM commands the Xen hypervisor in order to introduce the specified virtual machine. By introducing a virtual machine, we mean creating, instantiating and launching the virtual machine (and its content).

Sixth, Cluster Scaler is informed that the requested virtual machine has correctly been instantiated and is now in the running state. Then, Cluster Scaler reconfigures the Apache Load Balancer in order to take into account the new virtual machine (and its content) just introduced.

Finally, the load induced by the end users requests is now dispatched between the two virtual machines (containing the JOnAS Servers and the services).

VIII. VALIDATION

This section presents details, screenshots, and results about the demonstration associated to the scale-up use case.

First, our solution has been demonstrated to CELTIC and French National Research Agency (ANR) experts during the Servery project's mid-term review (the 7th of May 2010).

This live demonstration and the validation were done on three standards servers: one dedicated to the marketplace platform, one containing the marketplace management platform, and one in charge of injecting the end users load to the marketplace platform.

Over this hardware configuration, we observed that our whole MAPE loop runs approximately in 10 minutes (this is an average value coming from ten consecutive experimentations. These 10 minutes are broken down as follows:

- 1 minute is taken by SLC and JASMINe monitoring in order to monitor and detect 5 consecutive SLA violations in a 1 minute sliding time slot.
- 1 minute is taken by JASMINe monitoring for the planning of the scale-up action and the launching of the execution step.
- 1 minute is spent by JASMINe VMM in order to interact with the Xen hypervisor for introducing a new virtual machine.
- At least 6 minutes are consumed by the creation, the boot and the initialization steps of the (just introduced) virtual machine.
- Less than 1 minute is spent by Cluster Scaler to reconfigure the marketplace platform and check its state.

Note that the creation of the virtual machine can be reduced to a dozen seconds via the use of virtual machine templates; the boot and initialization steps can't be easily shortened.

Figure 5 below is a screenshot of SLC. It shows SLC results: here, one violation of the SLA tsla_id_3 has been detected).

MsgId	OccuredAt	jVM_freePhysicalMemorySize (octets)	jVM_CpuLoad	JOnAS state	appli_numberOfRequests	appli_averageProcessingTime	SlaId	SLCResult
uuid:92	2010-05-06 18:40:06.218	167325696	0.00396275	j2ee.state.running	0	-1.0	tsla_id_3	GREEN
uuid:91	2010-05-06 18:40:01.187	167325696	0.0039753527	j2ee.state.running	0	-1.0	tsla_id_3	GREEN
uuid:90	2010-05-06 18:39:56.14	167325696	0.0079507055	j2ee.state.running	0	-1.0	tsla_id_3	GREEN
uuid:89	2010-05-06 18:39:51.125	167325696	0.00385505	j2ee.state.running	0	-1.0	tsla_id_3	GREEN
uuid:88	2010-05-06 18:39:45.937	167325696	0.0059265113	j2ee.state.running	0	-1.0	tsla_id_3	GREEN
uuid:87	2010-05-06 18:39:40.875	167325696	0.003950227	j2ee.state.running	0	-1.0	tsla_id_3	GREEN
uuid:86	2010-05-06 18:39:35.812	167317504	0.0039635357	j2ee.state.running	0	-1.0	tsla_id_3	GREEN
uuid:85	2010-05-06 18:39:30.765	167301120	0.0059253406	j2ee.state.running	0	-1.0	tsla_id_3	GREEN
uuid:84	2010-05-06 18:39:25.703	167325696	0.00396275	j2ee.state.running	0	-1.0	tsla_id_3	GREEN
uuid:83	2010-05-06 18:39:20.64	167325696	0.0059265113	j2ee.state.running	0	-1.0	tsla_id_3	GREEN
uuid:82	2010-05-06 18:39:15.593	167325696	0.00396275	j2ee.state.running	0	-1.0	tsla_id_3	GREEN
uuid:81	2010-05-06 18:39:10.546	167309312	0.0118156755	j2ee.state.running	17	1.0	tsla_id_3	GREEN
uuid:80	2010-05-06 18:39:05.484	167309312	0.07900454	j2ee.state.running	169	1.0946746	tsla_id_3	RED
uuid:79	2010-05-06 18:39:00.406	167309312	0.053333335	j2ee.state.running	186	1.0645162	tsla_id_3	GREEN

Figure 5. Screenshot of SLC with a SLA violation.

Figure 6 is a screenshot of JASMINe VMM. It shows the marketplace platform after a self-scale-up. Three virtual machines are displayed: one containing the Apache LB (called apache) and two containing each a JOnAS server and the Telco services (called jonasWorker1 and jonasWorker3).

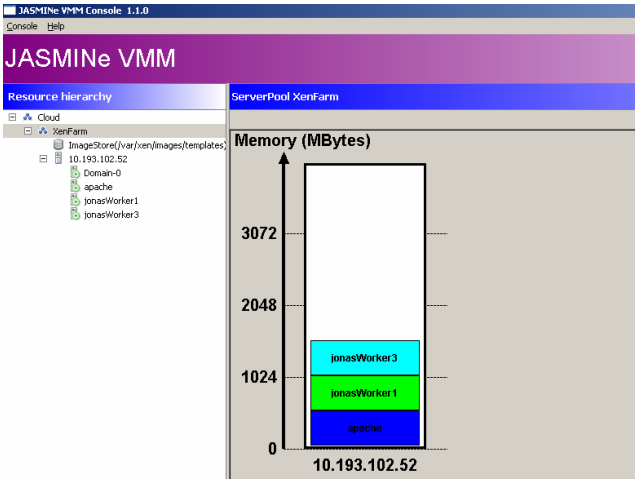


Figure 6. Screenshot of JASMINe VMM with 2 JOnAS servers.

Figure 7 below shows the number of requests, the average processing time, and the CPU load corresponding to jonasWorker1. Here, we have injected two identical loads on the Apache LB. The first load has led to a SLA violation and

the marketplace platform has been self-scaled-up. The second load has been injected after the self-scale-up action; the load is now balanced between the two jonasWorkers.

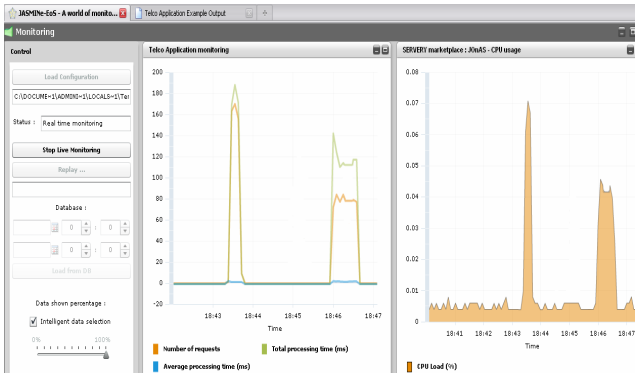


Figure 7. Screenshot of JASMINe Monitoring graphs.

IX. CONCLUSION

In this paper, we have presented an innovative open-source solution for self-scaling the cloud to meet service level agreements. Our solution has been applied to the Cloud Computing context via a self-scaling use case coming from the European CELTIC Servery cooperative research project. Applying our proposal to this use case has led us to several conclusions. First, according to the objectives, it allows to self-scale a virtualized cloud depending on the compliance with SLA. It also allows separating concerns related to the monitoring, analyzing, planning and executing steps in an industrial context and in the frame of an industrial use case.

Second, our solution is functional and efficient. It has been demonstrated in front of experts and validated.

Third, one of the important challenges we solved with this solution was to find, extend/modify, and integrate open-source middleware pieces with respect to industrial constraints raised by our R&D centers.

Last, but not least, this solution is well accepted by both France Telecom and Bull production project teams.

As future work, we consider to work on the Servery self-scale-down and self-repair use cases. We also plan to introduce several monitoring probes in the marketplace platform, to extend the SLC module in order to check more complex SLAs, and to embed it in JASMINe monitoring in order to take advantage of its monitoring mechanisms. We also plan to extend both the Drools rules for the analysis step and the planning mechanism in order to handle the two remaining use cases. We also wish to use JASMINe VMM capabilities in order to test our solution on a VMware marketplace platform.

ACKNOWLEDGMENT

We would like to thank the European CELTIC program for co-funding the SERVERY project (project number CP5-023), as well as both France Telecom - Orange Labs and Bull. Special thanks to Dr. Alexandre LEFEBVRE for his help and to Dr. Thierry COUPAYE for hosting this work.

REFERENCES

[1] IBM, "An architectural blueprint for autonomic computing", white paper, http://www-01.ibm.com/software/tivoli/autonomic/pdfs/AC_Blueprint_White_Paper_4th.pdf, Jun. 2006, [last accessed Nov. 2010].

[2] Horn P., "Autonomic Computing: IBM's perspective on the State of Information Technology", in IBM corporation, http://www.research.ibm.com/autonomic/manifesto/autonomic_computing.pdf, Oct. 2001, [last accessed Nov. 2010].

[3] IBM, "Simplified Policy Language", http://download.boulder.ibm.com/ibmdl/pub/software/dw/autonomic/ac-spl/ac-spl-pdf.pdf, 2008, [last accessed Nov. 2010].

[4] IBM, "Virtualization Management", http://www-01.ibm.com/software/tivoli/solutions/virtualization-management/, 2010, [last accessed Nov. 2010].

[5] Oracle, "Monitoring Performance Using the WebLogic Diagnostics Framework", http://www.oracle.com/technetwork/articles/cico-wldf-091073.html, August 2009, [last accessed Nov. 2010].

[6] Oracle, "Service Management", http://download.oracle.com/docs/cd/B19306_01/em.102/b31949/service_management.htm, 2009, [last accessed Nov. 2010].

[7] Oracle, "Oracle Enterprise Management 11g Database Management", http://www.oracle.com/technetwork/oem/db-mgmt/index.html, 2010, [last accessed Nov. 2010].

[8] Oracle, "Oracle VM Management Pack", http://www.oracle.com/technetwork/oem/grid-control/ds-ovmp-131982.pdf, 2010, [last accessed Nov. 2010].

[9] Oracle, "Platform-as-a-Service Private Cloud with Oracle Fusion Middleware", Oracle White Paper, http://www.oracle.com/us/036500.pdf, October 2009, [last accessed Nov. 2010].

[10] K. Dias, M. Ramacher, U. Shaft, V. Venkataramani, and G. Wood, "Automatic Performance Diagnosis and Tuning in Oracle", 2nd Conference on Innovative Data Systems Research (CIDR), http://www.cidrdb.org/cidr2005/cidr05cd-rom.zip, pp. 84-94, 2005.

[11] Eurescom, " Autonomic Computing and Networking: The operators' vision on technologies, opportunities, risks and adoption roadmaps", http://www.eurescom.eu/~pub/deliverables/documents/P1800-series/P1855/D1/, 2009, [last accessed Nov. 2010].

[12] Servery consortium, "SERVERY Celtic project", http://projects.celtic-initiative.org/servery/, 2010, [last accessed Nov. 2010].

[13] Chazalet A., "Service Level Agreements Compliance Checking in the Cloud Computing", 5th International Conference on Software Engineering Advances (ICSEA), pp. 184-189, 2010.

[14] OW2 consortium, "JASMINe Monitoring", http://wiki.jasmine.ow2.org/xwiki/bin/view/Main/Monitoring, 2010, [last accessed Nov. 2010].

[15] JBoss Community, "Drools 5 - The Business Logic integration Platform", http://www.jboss.org/drools, 2010, [last accessed Nov. 2010].

[16] OW2 consortium, "JASMINe Virtual Machine Management", http://wiki.jasmine.ow2.org/xwiki/bin/view/Main/VMM, 2010, [last accessed Nov. 2010].

[17] OW2 consortium, "OW2 JOnAS open-source Java EE 5 Application Server", http://jonas.ow2.org/, 2010, [last accessed Nov. 2010].

[18] Citrix Systems, "The Xen Hypervisor", http://www.xen.org/, 2010, [last accessed Nov. 2010].

[19] OW2 Consortium, "JASMINe: The Smart Tool for your SOA Platform Management", http://jasmine.ow2.org/, 2010, [last accessed Nov. 2010].

 ISBN: 978-1-61208-106-9

Exploitation of Vulnerabilities in Cloud Storage

Narendran Calluru Rajasekar
School of Computing and Technology,
University of East London,
London, U.K.
crnarendran@gmail.com

Chris Imafidon
School of Computing and Technology,
University of East London,
London, U.K.
chris12@uel.ac.uk

Abstract - The paper presents the vulnerabilities of cloud storage and various possible attacks exploiting these vulnerabilities that relate to cloud security, which is one of the challenging features of cloud computing. The attacks are classified into three broad categories of which the social networking based attacks are the recent attacks which are evolving out of existing technologies such as P2P file sharing. The study is extended to available defence mechanisms and current research areas of cloud storage. Based on the study, simple cloud storage is implemented and the major aspects such as login mechanism, encryption techniques and key management techniques are evaluated against the presented attacks. The study proves that the cloud storage consumers are still dependent on the trust and contracts agreed with the service provider and there is no hard way of proven defense mechanisms against the attacks.

Keywords-cloud storage; security; architecture; vulnerabilities

I. INTRODUCTION

Computers has evolved from small computing devices such as abacus to super computers, and the computing has changed from stand alone computers to centralised computing and then to distributed computing. Current era is cloud computing era where all the software, platform and infrastructure are virtualised and provided as services typically exploited using pay-per-use model. In traditional model, a company or an organisation will maintain their own Information Technology Infrastructure and hence had full control on the data and processes. But in cloud computing, the data and processes are maintained by some 3rd party vendors and hence the control is lost.

Internet is ubiquitous and its penetration rate is very high in recent years. 82.5 percent of United Kingdom residents have access to Internet [28]. It is the communication channel for cloud computing and almost everyone using Internet is involved in some form of cloud computing activities such as Gmail, Yahoo, Picasa, Facebook and so on. Since it is open to everyone, if a soft spot is identified by attackers, it could be exploited to a great extent. Hence it is the tempting target for cyber crime.

Most of the companies like to move their applications to cloud services due to the huge cost savings it provides. But they are taken aback due to one main reason "Security". There are many leading cloud storage service providers such as Google Docs, Amazon Simple Storage Service (Amazon S3), Nirvanix, Adrive and Zumo drive. The vulnerabilities of cloud storage are very high, even the leading service providers have been compromised at some point.

In this paper, under the title "Attacks", various vulnerabilities in cloud storage are identified that could be exploited. "Implementation" discusses the solution implemented to counter the attacks based on the study. Finally, implementation is evaluated and conclusion is drawn.

II. ATTACKS

The cloud storage attacks can be classified into three broad categories network / resource based, browser based and social networking based attacks.

A. NETWORK / RESOURCE BASED

1) Denial of Service

It is the most popular attack in Network Security where the user is abstained from getting the normal service from the service provider with the help of other attacks such as Internet Control Message Protocol (ICMP) Flood, SYN Flood, User Datagram Protocol (UDP) Flood and Smurf attack. It can also be performed by increasing load on Central Processing Unit (CPU), primary memory, network to slow down or eventually crash the system. Distributed Denial of Service (DDoS) attack is based on DoS, which consists of three layers, controller layer, broker layer and attacker layer. In DDoS, the actual attack is made from the broker layer by receiving commands from the controller layer. Since it involves different layers attacker information can be hidden easily [29].

Since the attack can be performed using various other attacks, both detection and prevention steps can be taken. Lee et al. [7] has proposed a DoS/DDoS intrusion detection system, which uses cumulative sum algorithm to detect the attacks. Kompella et al. [22] proposed a novel data structure called partial completion filter, which can detect claim and hold attack, which is not handled in the former system.

2) Buffer Overflow

Buffer Overflow is the most common vulnerabilities for past two decades where an attacker seeks for partial or total control of a host. This is caused when exceptions are not handled properly. For example: out of bound exceptions and type exceptions when not handle properly can be exploited to move the control to a function introduced by an attacker and the results are endless [3].

The buffer overflow vulnerabilities can be avoided by taking little extra care during development of software.

 ISBN: 978-1-61208-106-9

Further down, buffer overflow can be prevented even at the Kernel level [30] and Hardware level [32] as well.

3) Virtual Machine Based Rootkit

Virtualisation is an important aspect of cloud computing where the software, operating system and all related components are packaged together such that it is independent of the hardware [15]. This is facilitated by multiplexing the system with a small privileged kernel known as hypervisor. Virtual Machine Based Rootkit (VMBR) is a new type of malware, which is similar to hypervisor, installs underneath the operating system layer and hoist the operating system to virtual machine. Hence, it is difficult to detect VMBR's state by the software running on the operation system. Vitrol and Subvert are other rootkits that use this technique [25]. VMBR allows other malicious software or services to run on it, which is protected by the operating system. According to King et al. [25], the best way to detect VMBR is to control the layer beneath it with the help of a secure hardware or bootable media.

4) Side Channel

Although complex cryptographic algorithms are devised for security, the weakness in implementation is exploited to break the security. The side channel attack exploits the unintended data leakage such as power consumption, timing information to break the keys [19].

Lee et al. [11] proposed Lock and Key technique, which includes a test security controller that randomises the sub chains when accessed by an unauthorised user and hence reducing the predictability. An attacker has to break many layers of security in order to the access the scan chain in order to exploit it.

5) Man in the Middle

There are different variants of man in the middle attack exists. One of the variants is where an attacker having a device with two wireless cards can launch this attack. First he sends de-authentication frames of legitimate user to the service station. Legitimate user will be disconnected from the service station and start searching for access point in the same channel. Now attacker can use one of his wireless cards to act as service station and connect the legitimate user, mean while use the other wireless card to get connected to the actual service station using legitimate user Media Access Control (MAC) address [23]. Thus man in the middle attack can be launched successfully.

Once the attack is successfully launched, attacker can even attack Hyper Text Transfer Protocol Secure (HTTPS) communication. In this scenario, the user will be displayed a security certificate warning, which most of the users ignore and hence the system is compromised [6].

6) Replay Attack

Replay attack is where an attacker is able to capture the network traffic and replay it at a later time to gain access to the unauthorised resources even if it is encrypted. It has more effect on the Dynamic Rights Management (DRM) content where the rights over the resource changes over the time or number of times/ bandwidth used. If the attack is on the DRM content, the user not only looser privacy but also cost involved in the dynamic rights. For example, if user is accessing a file from the cloud storage service such as Amazon S3 where he is charged based on the amount of data transferred, the replay attack will eat up user's money. Abbadi et al. [9] have proposed a solution against the replay attack, which give users flexibility to use and manage the DRM content in any of the devices they own.

7) Resource Exhaustion

"Resource-exhaustion vulnerability is a specific type of fault that causes the consumption or allocation of some resource in an undefined or unnecessary way, or the failure to release it when no longer needed, eventually causing its depletion. [10]"

As stated in the definition, resource exhaustion vulnerability can be exploited to cause Denial of Service (DoS) attacks. This can be caused by bad design or inefficient utilization of resources on the service side and resource leakage where the resources are not released or destroyed from memory after use [10]. Hence it is difficult to observe and identify the cause unless it is closely monitored.

Antunes et al. [10] has proposed methodology to detect resource exhaustion vulnerability using which implemented Predator, a black box testing tool to identify the resource exhaustion vulnerabilities of a system. The operations of the tool involve attack generation and injection campaigns. Incorporating this methodology in software development life cycle (SDLC) can reduce this vulnerability.

8) Byzantine Failure

In cloud storage, many nodes participate to complete an activity. For instance there could be many redundant servers involved and also multiple users accessing single source. In this scenario, any of the nodes i.e., servers or users participating can fail arbitrarily as a result of crash or malicious activity, which is known as Byzantine failure [13]. System can be made robust by implementing threshold cryptography [2] to ensure the system is tolerant to Byzantine failure.

Recently, Wang et al. [24] proposed Agreement Protocol for cloud computing (APCC), which involves two processes Interactive Consistency Process and the Agreement Process. The Interactive Consistency Process is executed at the server nodes, which shares and stores the initial message among them. The server nodes then aggregate the results and transmit the message to client nodes. The Agreement Process is executed at the client nodes to receive the agreed message.

B. BROWSER BASED

1) XSS

Cross Site Scripting (XSS) can be used to inject malicious code into the client machine by exploiting the client side script vulnerability in the website [1]. Thus an attacker can introduce his own script and impersonate user credentials to perform malicious activity in the website such as session Hi-jacking and also craft phishing sites. A user called Samy added more than 1 million buddies to his My Space account by exploiting XSS Vulnerability [5]. Even Google has suffered from XSS vulnerability in their online spreadsheet application using which the user cookie can be stolen, which is valid for other sub-domains also. From the server point of view detecting and preventing XSS attack is a

 ISBN: 978-1-61208-106-9

difficult task as it is done at the client end for which server has not much control. Wurzinger et al. [20] has proposed a server side solution SWAP (Secure Web Application Proxy) to detect and prevent XSS. It has a reverse proxy, which intercepts the HTML response from client and validates it for XSS attack.

2) SQL Injection

Similar to XSS, SQL Injection is also a vulnerability, which can be used to inject malicious database scripts when user inputs are not properly validated. This can generally be prevented by passing user inputs as parameters and avoiding query building based on the user input. At times there will be scenarios where building queries based on user inputs are unavoidable. In these cases, vulnerable user input validation must be performed to prevent SQL Injection. SQL Injection is very dangerous as it can be used to change values of multiple records and can even be used to delete the whole table [31].

For example, consider the following SQL statement, which updates a value based on email id.

UPDATE [User Information] SET [Credit] = '£1000000' WHERE [Email Id] = '$email'

If user passes value ['xyz@abc.com' OR 'x'='x] for $email. Then 'x'='x' condition is always true and hence credit will be set to '£1000000' for all users.

3) Malware

A program designed to damage the machine is called malware [14]. The web browsers are more susceptible to malwares as it supports extension of 3rd party programs through "Add-on" or "Plug-in" capability. Louw et al. [17] has demonstrated a malware program, which is capable of capturing sensitive information such as passwords even when the communication is done using Secure Socket Layer (SSL). This is possible because the information is captured even before the communication begins when the data is encrypted for submission. Some of the Internet security program may detect and warn some of the malicious activities of malware programs such as submitting hidden data to remote server, but not many users are not aware of the technical details and tend to ignore the warning.

4) XML Wrapping

Web Services is a key technology to implement Service Oriented Architecture (SOA) especially is very useful for implementing interoperable and platform independent services. Extensible Markup Language (XML) is the underlying mark up language used to communicate between server and client. XML signatures facilitate the unauthorised modification and origin authentication for the XML documents [16].

A SOAP message with a signed body can be moved to different wrapper message without altering the signatures. Hence the resulting SOAP message is still valid producing valid hash [18]. This is called XML Wrapping attack, which according to Gruschka et al. [18] can be mitigated using SOAP message security validation and XML schema validation but the formal proof of safety is missing.

C. SOCIAL NETWORK

1) Sybil Attack

In a Sybil attack [8], a malicious user acquires multiple identities and pretends to be distinct users and tries to create a relationship with honest users. Even if one honest user is compromised, malicious user will gain special privileges, which can be used for attacks. Cloud storage is widely used in social networking such as Facebook, My Space, Orkut, Bebo where users can store their files such as documents, photos and videos and share it easily with their network. The relationship between the honest user and the malicious user is called attack edge, which can even be used for social engineering.

Vanish [21] is a proposed system, which increases privacy by self destructing data. Using this system, a message can be encrypted using a random key, which is stored in the distributed hash table (DHT). These keys will be destructed from DHT after user specified interval and hence the data is lost forever. User can de-encrypt the data using the key before it is destructed forever. This seems to be a solution for P2P based storages and also has been simulated for Gmail using Firefox Plug-in but in its current form it is not adequately protected against Sybil attacks and can be defeated [26].

2) Social Intersection Attacks

Social Intersection attacks can be effectively launched in social networking environment. It can be used to identify the original owner of the shared anonymous data object with just two compromised users in a group [12]. It is hard to detect this kind of attack as it is performed passively and become more powerful with increase in number of compromised users. A solution is proposed for this attack where the service provider can build a number of anonymous nodes around a user and hence highly reducing the probability of identifying the originating source.

3) Collusion Attack

Secret Key Multiplication (SKM) group re-keying scheme is used in group collaboration, where multiple users participate in a group discussion that might involve sharing of various resources such as text, files and even hardware resources. According to this scheme a subset key is generated for each group from a master key and in turn each user in a group is given a private key. It is assumed that the users in each group keep their key secret. Collusion attack is performed by combining information among two or more users and gain access to resources that the attacker is not supposed to have. Using collusion attack, it is possible to obtain even the high level key, which will give attacker access to other groups, which is not related to the attacker. Raphael [4] proved SKM group re-keying scheme to be vulnerable to collusion attacks.

III. IMPLEMENTATION

Tb drive [27], an online storage is implemented using the architecture devised based on the study. As a student it is hard to avail access to storage servers used for commercial purpose. Hence storage provided by a web hosting service was utilised to implement this project, which had limitations on server capacity and performance.

 ISBN: 978-1-61208-106-9

The application is developed using ASP.Net 3.5 with C# as server side script language, AJAX and Visual Studio 2010 IDE. MS SQL 2008 is used for storing data. IIS 7.0 is used as the web server to handle client requests and the developed application is tested using different browsers namely Fire Fox, Internet Explorer, Google Chrome and Safari under Mac OS X and Windows platforms.

This web application is designed to meet the functionality and security requirements, based on the analysis done in previous chapters. ASP.Net application service is used to implement the access controls for the web application.

Folders are stored in database using Hierarchyid, new data type available in SQL Server 2008. This facilitated displaying folder structure in Tree View control reducing number of Lines of Code (LOC).

The implementation of the application is discussed below in four major aspects namely login mechanism, storage organisation, file encryption and decryption and key management.

A. Login Mechanism

Tb drive doesn't store password to authenticate users, instead implements Open ID mechanism to authenticate user. Implementation accepts Google and Yahoo Open Ids for login to Tb drive and demonstrates simplest account creation process. Users can start using the application readily and securely without having to fill out sign up forms. User simply has to click on the Open ID provider logo. Application will be redirected to Open ID service provider. Open ID service provider will ask for user name and password for authentication mentioning the requesting domain (techbizarre.com) name in the authentication page.

Once the user enters valid user name and password, Open ID service provider will display all the details requested by the relying party (techbizarre.com) requesting approval from the user. If the user approves the details to be shared, the requested information will be sent to relying party. User can also choose to remember the association; hence this step can be skipped in future.

Open ID provides only authentication and don't support user session that has to be taken care by the relying party. Single-Sign-On generally known as SSO is another mechanism, which facilitates even the session to be taken care by the service provider and have its own advantages and disadvantages. Though Open ID doesn't facilitate session management, the service is provided free of charge where as cost is involved to avail SSO facility form 3rd party. If Open Id is implemented, application can accept services from many service providers and hence more audience where as when SSO is implemented, the service is restricted to one service provider.

B. Storage Organisation

In cloud storage, files can be stored in two different ways file system and database. Both has advantages and disadvantages; the file system requires full permission on the disk, which is difficult in a web hosting scenario than maintaining own server. The storage system requires a binding between the application layer and storage layer, which is loosely coupled when the files are stored in the file system and there is no concrete relationship between the files and the application layer where privacy and confidentiality is driven. Figure 1 depicts typical high level cloud storage architecture.

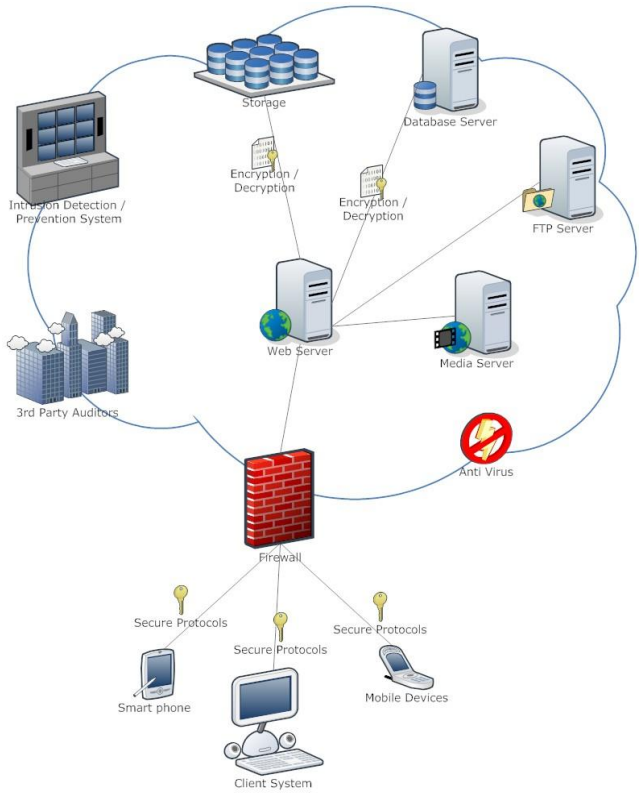


Figure 1. High Level Cloud Storage Architecture.

The binding between the application layer and storage layer can be made strong when the storage is implemented in database server. This may hinder other support such as File Transfer Protocol (FTP) services for the storage facility.

Considering the web hosting storage scenario of Tb drive, database is chosen for storage layer but storage using File system is also demonstrated without the feature of encryption.

Hierarchyid data type available in SQL Server 2008 is used to implement the folder structure in Tb drive, which allows easy mapping of folder structure to tree view control.

C. File Encryption & Decryption

Since the files are stored in 3rd party server, this might break user's privacy. Hence the ability to encrypt files can be provided to users. In security perspective, submitting encrypted files seems more appropriate; but, when considering key management, there exists risk of user using various keys to encrypt files and may get lost when retrieving data. To make it more user friendly, Tb drive uses master key concept which is explained under Key Management. Tb drive accepts a key string from user, which is internally converted to key and vector combination to

 ISBN: 978-1-61208-106-9

encrypt files, Tb drive demonstrates symmetric encryption of files using Rijndael encryption algorithm. Individual files can be encrypted using a key. Since symmetric encryption is used, user has to provide the same key to decrypt the files. If the key is lost, the data is lost forever.

D. Key Management

Since loss of key can lead to data lost forever, key management is crucial in cloud storage. To mitigate this risk, Tb drive stores one way hashed master key supplied by user in the database and uses the master key to encrypt files. Since Tb drive only stores hashed master key, unless user provides the master key data cannot be decrypted. Thus users can safely store data in Tb drive having full access with them. The other concern here is that the law-maker can be law-breaker, i.e., the key management technique logic is again provided by the 3rd party service providers who can internally change the logic. Hence the service provider should conduct regular external audit that can verify and certify the credibility of the application and thus it can be trusted.

IV. CONCLUSION AND FUTURE WORK

People are addicted to simplicity and are lazy to manage multiple passwords. One of the solutions to overcome this could be Open ID, which increases usability but at the same time increases the risks of XSS and phishing attacks. Hence general awareness is required on Do's and Don'ts of cloud computing.

ASP.Net Membership service simplified the implementation of session security and assumed to be safe from attacks. If any of the vulnerability in the Membership assembly is exploited then the assumption is flawed.

Simple flaws in coding such as not destroying objects in memory that is no longer required can be easily exploited to launch DoS attack and buffer over flow attack, which decrease the performance of the system or even crash the server. Coding flaws such as lack of input validations can lead to SQL injection through which an attacker can gain access full database.

Service provider has full access and in most of the cases they have ability to impersonate a customer and have full access over his data. An attack by a disgruntle employee of the service provider can easily break in to consumers data. Encryption can protect sensitive data but still vulnerable to Man in the Middle attack.

Attacks through social networks are recent ones, which are being exploited as this doesn't require any extra burden for an individual to initiate. Most of the social network users are naive enough to give away sensitive and personal information in social network websites, which can then be used to break the password using password recovery facility that every service provider provides.

A famous phrase exists "Words spoken cannot be taken back"; similarly Data given away to cloud cannot be taken back. No one knows how many backup exists for the data stored in cloud. A user can upload unencrypted data assuming to encrypt it after uploading into the cloud. Even before the data is encrypting, it could have been backed up and user is left unaware.

Cloud service relies on the network, which is not always secure. Network sniffers can easily gain sensitive information by monitoring network payloads.

Security should be implemented at each and every layers defined in International Standards Organisation – Open System Interconnect (ISO-OSI) Model and at very granular level. The attacks such as denial of service attack, buffer overflow attack, man in the middle attack exists for ages, still there is no concrete mechanisms to counter these attacks. Even the strongest encryption available is vulnerable to side channel attacks.

Even the leading service such as Google, Zoho, Nirvanix has failed at some point exposing customer data and even losing the data. Hence users should have their own backup mechanisms for critical data.

ACKNOWLEDGEMENT

The success of this work largely depends on the motivation and encouragement provided by all my tutors in the University of East London. I also extend my thanks to all my friends who have been supportive achieving this work and special thanks to Arul Arasu Ganesan, Brindha Sheshadri and Madhuvanthi Dayalan who reviewed this work.

I also owe a big thanks to University of Cambridge for the security seminar held at William Gates building. This helped me to learn aspects of security oriented languages.

I owe a special thanks to MSDN Academic Alliance, which made possible for me to gain access to recent version of Microsoft Window Server 2008, Visual Studio 2010 and Microsoft SQL Server 2008.

REFERENCES

[1] A. Kieyzun, P. J. Guo, K. Jayaraman, and M. D. Ernst. "Automatic creation of SQL injection and cross-site scripting attacks", Proceedings of the 2009 IEEE 31st International Conference on Software Engineering, pp. 199-209, 2009.

[2] C. Cachin and S. Tessaro. "Optimal resilience for erasure-coded Byzantine distributed storage." Distributed Computing, pp.497-498, 2005.

[3] C. Cowan, P. Wagle, C. Pu, S. Beattie, and J. Walpole. "Buffer Overflows: Attacks and Defenses for the Vulnerability of the Decade," oasis, pp.227, Foundations of Intrusion Tolerant Systems (OASIS'03), 2003.

[4] C. Raphael. "Collusion attacks on secret keys multiplication (skm) group re-keying scheme proposed at CITA03.", unpublished.

[5] E. Levy and I. Arce. "New threats and attacks on the world wide web." IEEE Security & Privacy, pp. 234-266, 2006.

[6] F. Callegati, W. Cerroni, and M. Ramilli. "Man-in-the-Middle Attack to the HTTPS Protocol," IEEE Security and Privacy, vol. 7, no. 1, pp. 78-81, Jan./Feb. 2009, doi:10.1109/MSP.2009.12

[7] F. Leu and Z. Li. "Detecting DoS and DDoS Attacks by Using an Intrusion Detection and Remote Prevention System", IEEE Conference and Exposition, pp. 1-15, 2009.

[8] H. Yu, M. Kaminsky, P. B. Gibbons, and A. Flaxman. "SybilGuard: Defending Against Sybil Attacks via Social Networks." IEEE/ACM TRANSACTIONS ON NETWORKING, vol. 16, no. 3, pp. 576-589, 2008.

[9] I. M. Abbadi and M. Alawneh. "Replay Attack of Dynamic Rights within an Authorised Domain," securware, pp.148-154, 2009 Third International Conference on Emerging Security Information, Systems and Technologies, 2009.

[10] J. Antunes, N. F. Neves, and P. J. Ver. "Detection and Prediction of Resource-Exhaustion Vulnerabilities" issre, pp.87-96, 2008 19th International Symposium on Software Reliability Engineering, 2008.

[11] J. Lee, M. Tehranipoor, C. Patel, and J. Plusquellic. "Securing Designs against Scan-Based Side-Channel Attacks." IEEE transactions on dependable and secure computing, vol. 4, no. 4, pp. 325-336, 2007.

[12] K. Puttaswamy, A. Sala, and B. Y. Zhao. "StarClique: guaranteeing user privacy in social networks against intersection attacks", Proceedings of the 5th international conference on Emerging networking experiments and technologies, pp. 157-168, 2009.

[13] K., Driscoll, B. Hall, H. Sivencrona, and P. Zumsteg. "Byzantine fault tolerance, from theory to reality." Computer Safety, Reliability, and Security, vol. 2788, pp. 235-248, 2003, doi: 10.1007/b12002.

[14] M. D. Preda, M. Christodorescu, S. Jha, and S. Debray. "A semantics-based approach to malware detection.", ACM Transactions on Programming Languages and Systems (TOPLAS), vo. 30, no. 5, pp. 25, 2008.

[15] M. F. Mergen, V. Uhlig, O. Krieger, and J. Xenidis. "Virtualization for high-performance computing.", ACM SIGOPS Operating Systems Review, vol. 40, no. 2, pp. 11, 2006.

[16] M. McIntosh and P. Austel. "XML signature element wrapping attacks and countermeasures", Proceedings of the 2005 workshop on Secure web services, pp. 20-27, 2005.

[17] M. T. Louw, J. S. Lim, and V. N. Venkatakrishnan. "Enhancing web browser security against malware extensions." Journal in Computer Virology, vol. 4, no. 3, pp. 179-195, 2008.

[18] N. Gruschka and L. Iacono. "Vulnerable Cloud: SOAP Message Security Validation Revisited", IEEE International Conference on Web Services, pp. 625-631, 2009.

[19] N. R. Potlapally, A. Raghunathan, S. Ravi, N. K. Jha, and R. B. Lee. "Aiding side-channel attacks on cryptographic software with satisfiability-based analysis.", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 15, no. 4, pp. 465-470, 2007.

[20] P. Wurzinger, C. Platzer, C. Ludl, E. Kirda, and C. Kruegel. "SWAP: Mitigating XSS attacks using a reverse proxy", ICSE Workshop on Software Engineering for Secure Systems, pp. 33-39, 2009.

[21] R. Geambasu, T. Kohno, A. Levy, and H. M. Levy. "Vanish: Increasing data privacy with self-destructing data" Unisex Security Symposium, vol. 18 pp. 299-350, 2009.

[22] R. Kompella, S. Singh, and G Varghese. "On Scalable Attack Detection in the Network." IEEE/ACM TRANSACTIONS ON NETWORKING vol. 15, no. 1, 2007.

[23] R. Syahputri and M. Hasibuan. "Security in Wireless LAN Attacks and Countermeasures", SNATI, pp.54-78, 2009.

[24] S. C. Wang, K. Q. Yan, S. S. Wang, and C. P. Huang. "Achieving high efficient agreement with malicious faulty nodes on a cloud computing environment", Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human, pp. 468-473, 2009.

[25] S. King and P. Chen. "SubVirt: Implementing malware with virtual machines", IEEE Symposium on Security, pp. 1-14, 2006.

[26] S. Wolchok, O. S. Hofmann, N. Heninger , E. W. Felten , J. A. Halderman , C. J. Rossbach, et al. "Defeating Vanish with Low-Cost Sybil Attacks Against Large DHTs", Citeseer, 2010.

[27] Tb Drive. http://www.techbizarre.com/tbdrive/ seen: 29.08.2010

[28] United Kingdom Internet Usage Stats and Market Report. http://www.internetworldstats.com/eu/uk.htm seen: 25.08.2010

[29] W. Liu. "Research on DoS Attack and Detection Programming", IITA, pp. 207-210, 2009.

[30] W. Speirs. "Making the kernel responsible: a new approach to detecting & preventing buffer overflows", Proceedings of the Third IEEE International Workshop on Information Assurance, pp. 21-32, 2005.

[31] X, Fu and K. Qian. "SAFELI: SQL injection scanner using symbolic execution", Proceedings of the 2008 workshop on Testing, analysis, and verification of web services and applications, pp. 34-39, 2008.

[32] Z. Chen and X. Yan. "Hardware Solution for Detection and Prevention of Buffer Overflow Attacks in CPU Micro-architecture." RESEARCH AND PROGRESS OF SSE, vol. 26, no. 2 pp. 214-219, 2006.

 ISBN: 978-1-61208-106-9

Towards a Security Management Reference Model for Vertical and Horizontal Collaborative Clouds

Michael Kretzschmar and Sebastian Hanigk
Universität der Bundeswehr München, Institut für Technische Informatik,
Werner-Heisenberg-Weg 39, 85577 Neubiberg, Germany
{michael.kretzschmar, sebastian.hanigk}@unibw.de.

***Abstract*—The re-perimeterization and the erosion of trust boundaries already happening in organizations is amplified and accelerated by Cloud Computing. Security controls in Cloud Computing are, for the most part, no different from security controls in any IT environment from a functional perspective. However, because of the Cloud service models employed, the operational models, and the technologies used to enable Cloud services, Cloud Computing may present different risks and additional requirements to an organization than traditional IT solutions. This paper focuses on security management issues for vertical and horizontal Collaborative Clouds. Based on a detailed and comprehensive analysis of requirement domains, currently offered solutions, security management objects that have to be managed, integrated or adopted, we introduce a Cloud Security Management Reference Model (CSMRM), integrating various Cloud security services of an organization and providing interoperability to identified stakeholders. This new model adopts the Security Management Infrastructure (SMI) approach and establishes the basis for a global and consistent management of the Cloud security infrastructure according to organizational goals.**

***Keywords*-Security Management Infrastructure, Collaborative Clouds, Cloud Security Management Reference Model**

I. INTRODUCTION

Today, every new trend in the Information Technology (IT) has to face issues about security. Most current Cloud offerings are all over the map on the security issue, ranging from largely insecure installations for some commodity and private Cloud offerings to about half of the way towards meeting that goal for the best enterprise public Clouds [1]. One of the most important security challenges is to assure a predefined security level of trust over multi-provider Cloud Computing environments with dedicated communication infrastructures, security mechanisms, processes and policies [2]. Thus, it is necessary to overcome 'security islands' and vendor lock-in. Inadequate security management (in order to establish trust and prevent risks) can be the show stopper for ubiquitous Cloud Computing usage, as Cloud Computing services will multiply and expand faster than the ability of Cloud Computing consumers to manage or govern their usage [3]. Ubiquitous connectivity, the amorphous nature of information interchange, and the ineffectiveness of traditional static security controls which cannot deal with the dynamic nature of Cloud services require enhanced security approaches with regard to Cloud Computing [2], [4].

The aim of Security controls in Cloud Computing is, for the most part, no different than security controls in any IT environment from a functional security management perspective. The Security Management Infrastructure (SMI) approach of the EU [5], NATO [6], the USA [7], or the UK [8], includes security management capabilities such as Identity Management, Privilege Management, Metadata Management, Policy Management, and Crypto Key Management. These functional capabilities will be adopted for Cloud Computing usage [9]. However, the private and public sector needs an objective about how this new computing paradigm will impact organizations from a security management perspective, how it can be used with existing technologies, and the potential pitfalls of proprietary technologies that can result in a lock-in effect or limited choice. To overcome this situation, we present a Cloud Security Management Reference Model (CSMRM) that allows to manage Cloud security management services and to integrate Cloud Computing security management into the (pre-existing) SMI of the whole organization. This CSMRM addresses further challenges and bridges the gap between current insufficient Cloud security management approaches and future Cloud security management.

This paper is structured as follows: In Section II, we provide a detailed description of a collaborative scenario covering all deployment types and delivery models in order to identify Cloud security management relevant components, requirements, and interfaces. In Section III, we provide the results of the requirements analysis, which guides the evaluation of related work in Section IV and the identification of security management objects in Section V. Finally, we introduce the CSMRM in Section VI. Section VII summarises and concludes this paper.

II. SCENARIO

The communication and information infrastructures of private and public sector organizations that are collaborating according to agreed security policies are shown by a scenario in this section (visualized in Figure 1). This infrastructure is controlled and managed traditionally and

 ISBN: 978-1-61208-106-9

includes various security devices, services, and processes from various manufacturers in order to manage IT security capabilities. The organization constructs an internal Cloud Computing infrastructure upon the existing IT infrastructure using open-source or commodity software that includes the Cloud Security Management Infrastructure (CSMI).

A single Cloud (e.g., *PrC-A2* in Figure 1) comprises Cloud services (of one or all types) and additional control and management elements, such as Service Management, Security Management, Service Catalogue, or Collaboration Service. These Cloud services can be used by individuals (e.g., the members of branch *A2* within organization *A*, members of other organizations or external users), and can be a single service or comprise other Cloud services in order to provide the desired functionality.

Services of the same type (SaaS, PaaS, or IaaS) are referred as *horizontal* Cloud services. In many instances, Cloud computing service provider will provide a value-added service on top of another Cloud provider's service. For example, if a SaaS provider needs flexibility, it may be more cost-efficient to acquire necessary infrastructure from an IaaS provider rather than building it. These more complex and integrated services are termed *vertical* Cloud services.

A private Cloud of an organization *A* may contain several Clouds itself (e.g., the Cloud of a branch). The integrated Clouds of organization *A* can be deployed on different geographic locations and organizational branches, subsidiaries, and units. This Cloud can be seen as a *Collaborative Cloud*, even when it is assumed to be a private Cloud from the perspective of organization *A*. Note that a private Cloud service from organization *A* may consist of several Cloud services from partners or from a public Cloud service provider. For example, a Cloud storage service from the private Cloud (*PrC*) of organization *A* may use another Cloud storage service from *PrC-B*, together with a storage system of Cloud *PuC-I*, a file system, and a tape storage system of Cloud *PuC-II*, in order to provide a new compound Cloud service. A Collaborative Cloud may also offer Cloud services which use several other Cloud services, for example an Information service that interacts with various stakeholders of an inter-organizational project.

Clouds are connected via Intranet (private) or Internet (public) connections. In addition, Cloud service brokers—providers that offer intermediation, monitoring, transformation, portability, etc. between various cloud providers—can be used while building compound services. Managers can make intelligent and flexible decisions about what parts of their application loads runs internally and what parts externally. As a rule of thumb, computation-intensive Cloud services are better provided and used by public Cloud providers, but as dynamic security policies may define other risks for the transferred data, such a Cloud service could be moved and deployed back into the private Cloud.

III. Requirements analysis

This section defines the requirements for a unified and collaborative Cloud security management, based on, but not limited to, the scenario given in the previous section. There are several sources [2], [10]–[12] providing questions with regard to Cloud security management (e.g., 'How do I manage and control my security policies along the whole Cloud Service Life-cycle?'). Based on these questions we discovered and defined four domains for Cloud security management requirements to cluster identified requirements. These domains are (1) Security Management Functions, (2) Collaboration, (3) Integration of Security Management Objects, and (4) General Requirements.

A. Security Management Functions

In a hybrid private and public Cloud scenario there is a significant incremental risk if outsourced services to the public Cloud bypass the technical and administrativ controls. Customers are ultimately responsible for the security and integrity of their own data, even when it is held by a service provider [13]. Due to this complexity and opacity, policy enforcement becomes critical at all possible enforcement points. Since it is difficult to ascertain where data may be directed to it becomes imperative to encrypt all data, whether it is in motion or at rest. The biggest question here is key management (e.g., single key for all users, one key per user, multiple keys per user, etc.) [14]. The trend toward multiple service providers has the potential for creating an identity nightmare unless it is coordinated across all platforms. Each service will need to identify the user and may carry a number of user attributes including preferences and history. Federated identity solutions are necessary for service providers to standardize on mechanisms for sharing authentication, authorization and access (AAA) information with each other.

B. Collaboration

Collaborative Clouds are built upon private and public Clouds of various organizations. The following aspects are presented to highlight the requirement spectrum for security management. We probably won't know exactly where or in which country our information is hosted [13]. In addition, information in the Cloud is typically in a shared environment alongside data from other customers. Therefore inter-security management information exchange is necessary, where Cloud security management applications of two or more organizations share information. The use of standardized and non-proprietary protocols to communicate and exchange information between security capabilities will support this inter-organizational sharing of information to prevent vendor lock-in threat, resulting in problems with data transfer between Cloud vendors [15]. Furthermore distributed time zones have to be considered in order to support adequate

 ISBN: 978-1-61208-106-9

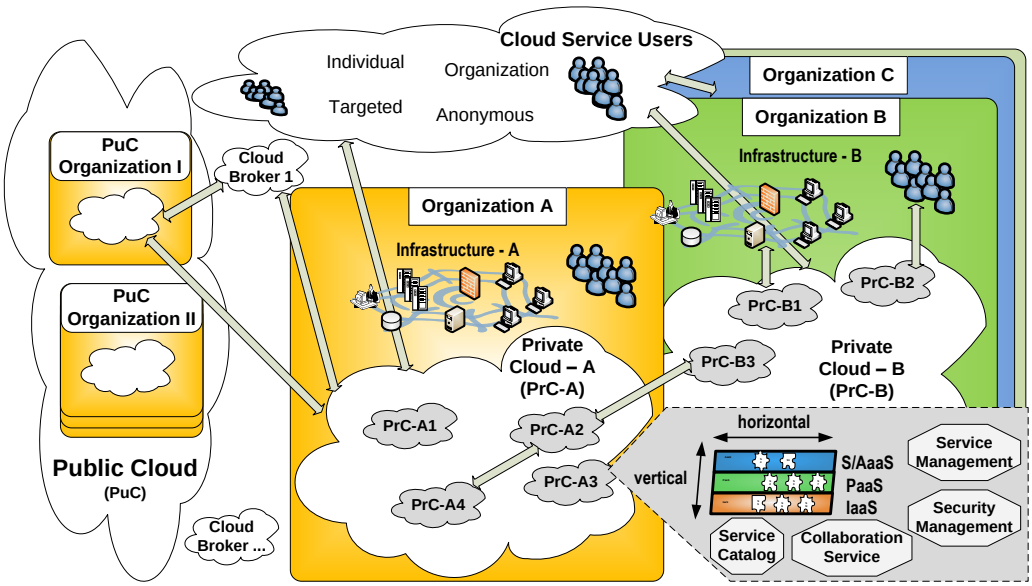


Figure 1. Vertical and Horizontal Collaborative Cloud scenario showing organization-specific private Clouds (*PrC-X*) and public Clouds (*PuC-X*).

timestamps (e.g., security auditing). But also from an intra-organizational perspective security management information exchange between the Cloud security management system and the overarching security management system of the whole organization has to be established in order to fulfill security policies comprehensively.

C. Integration of Security Management Objects

Cloud security management will allow a central operative management of all security capabilities. Therefore it has to provide interfaces and APIs in order to integrate all security-related data stored in the concrete security capabilities including Web-based ones. This will foster the move from manual to automated security management operations. The range of security capabilities that have to be considered by a security management depends mostly on the degree of integration and complexity of the provided services by the Cloud service provider. In the case of SaaS, this means that service levels, security, governance, compliance, and liability expectations of the service and provider are contractually stipulated; managed to; and enforced. In the case of PaaS or IaaS it is the responsibility of the consumer's system administrators to effectively manage the same, with some offset expected by the provider for securing the underlying platform and infrastructure components [2]. However to achieve this integration the adoption of a standards-based system design and implementation to enable interoperability, facilitate federated security management operations is necessary. This allows the use of commercial products, too.

D. General Requirements

As the number of Cloud services and their potential security management capabilities can get quite high in collaborative environments a scalable architecture is required. Additional ones may need to be configured if the collaboration grows or if components are replaced dynamically. But also the set of supported security management capabilities is not static. Continuously, new types of these capabilities evolve and manufacturers are extending their Cloud portfolio. In addition there is the need to operate in a multi-national or multi-cultural environment. Therefore the design and development of the system have to meet the requirements of a specific geographic or linguistic market segment.

IV. Related Work

The following section is structured into two parts. First we provide some theoretical foundations concerning security management models and cloud security areas. Secondly, we present an overview of current Cloud security management approaches and exchange standards.

The FCAPS model (ISO 10164) describes security management functions generically as goals in order to be implemented by security management tools. While the ISO/IEC 27001 offers a methodology and implementation guideline for providing and managing security services. [16] presents a Service Oriented Security Architecture (SOSA) as a collection of security services forming a security infrastructure used by Web Service providers. The Security Management Infrastructure approach, also known as Enterprise Security Management (ESM) will serve as an overarching security architecture integrating security capabilities (e.g., Identity,

 ISBN: 978-1-61208-106-9

Credential, Crypto Key Management, etc.) and managing them according to an organizational security policy [9]. There are several sources that describe Cloud Computing security areas [2], [14], [12], [1]. Unfortunately they differ in defining and covering necessary security management functional areas and collaboration aspects that can be used for a comprehensive Cloud security management. For example the management of meta-data or configuration management of security capabilities are not covered. Mainly they focus only to Identity, Privilege, Access and Crypto Key Management. The adaption and reuse of existing, traditional security management applications for Cloud Computing is proposed as one way-ahead [12]. A detailed summarization of fundamental characteristics and shortcomings of 14 of these security management systems are shown in Figure 2, which compares along a list of 5 design and 9 functional criteria c.q. requirements, indicates that neither of the observed approaches addresses the required range of security management functions, nor do they provide mechanisms for composed Cloud services [17].

Unfortunately only few applications c.q. models that focus unique to the Cloud security management aspect like Zscaler, Panda and the security management model [18] exist. But they only provide single security management function areas like policy-based secure web access or provides complete protection services. Furthermore there are services like PingIdentity, Symplified, etc. that covers the federated management of identities. Mostly there are some security management elements included within Cloud management applications. For example enStratus provides management for Amazon and The Rackspace Cloud infrastructure including security management functions like authentication and authorization, key management and audit. Further examples of Cloud management services like Scalr, Kaavo, CloudKlick, CloudStatus, RightScale, Elastra, Enomaly, Cloud42, etc. exist ([2], [14], [12]). DeltaCloud is an open source project aiming to develop an ecosystem of tools, scripts and applications for the Cloud. The project also aims to write a common, REST-based API to enable developers to write once and manage across multiple Clouds. One of the biggest challenges to Cloud Computing is the lack of standards as many efforts centred around the development of both open and proprietary APIs which seek to enable things such as management, security and interoperability for Cloud. Some of these efforts include the Open Cloud Computing Interface, Amazon EC2 API, VMware's DMTF-submitted vCloud API, Sun's Open Cloud API, Rackspace API, SNIA Cloud Data Management Interface (CDMI) and GoGrid's API, to name just a few. Beside these solutions, there are some standards and approaches for specific security areas. For the exchange of authentication and authorization data, standards as OASIS SAML, specifications of Liberty Alliance, and the Web Services Federation Language are implemented with their main focus on Web-based services. Furthermore in the security area of crypto key management the Key Management Interoperability Protocol (KMIP) can be used.

To summarize current Cloud security management approaches cannot fulfill all requirements put forward for an security management of Collaborative Clouds. The range of security fields supported is often limited and none of these tools are flexible and holistic enough to ensure the required level of interoperability and flexibility by implementing the presented Cloud standards.

V. Cloud Security Management Objects

There are significant trade-offs to each Cloud model in terms of integrated features, complexity vs. openness (extensibility), and security. The key takeaway for security management is that the lower down the stack the Cloud service provider stops, the more security capabilities and management consumers are responsible for implementing and managing themselves. However, there is still the question, what are the 'target objects' that have to be managed within the CSMI. In this section three domains for these objects are introduced that have to be addressed by a Cloud security management system.

A. Security functions provided by Cloud service providers

Various Cloud service providers add security functions covering also some parts of Cloud security management to their proprietary Cloud service offerings. For example Amazon Elastic Compute Cloud (Amazon EC2) Security supported a multi-factor authentication (knowledge and ownership) to gain access, control privileges and supporting of credentials like X.509 Certificate or proprietary Amazon Secret Access Key (e.g., to sign API calls). A key management allows the multiple concurrent usage of these certificates and keys. Beside that the access is logged and audited. Furthermore flexibility to place instances within multiple geographic regions as well as across multiple availability zones is possible, however the choice (e.g., region, continent) is limited [19].

B. Cloud security management services

PingFederate is a Cloud-based Identity-as-a-Service provider that focus in federating identity management and is integrated by a provider specific API. These kind of Cloud services can be classified as SaaS. The IdP (Identity Provider) sending identity attributes (from an authentication service or application) to PingFederate. PingFederate uses those identity attributes to generate a SAML assertion. PingFederate extracts the identity attributes from the incoming SAML assertion and sends them to the target application of a service provider as consumer of identity attributes. Initial user authentication is normally handled outside of the PingFederate. PingFederate offers integration kits (Windows IWA/NTLM, X.509 Certificate, LDAP Authentication Service), that access

 ISBN: 978-1-61208-106-9

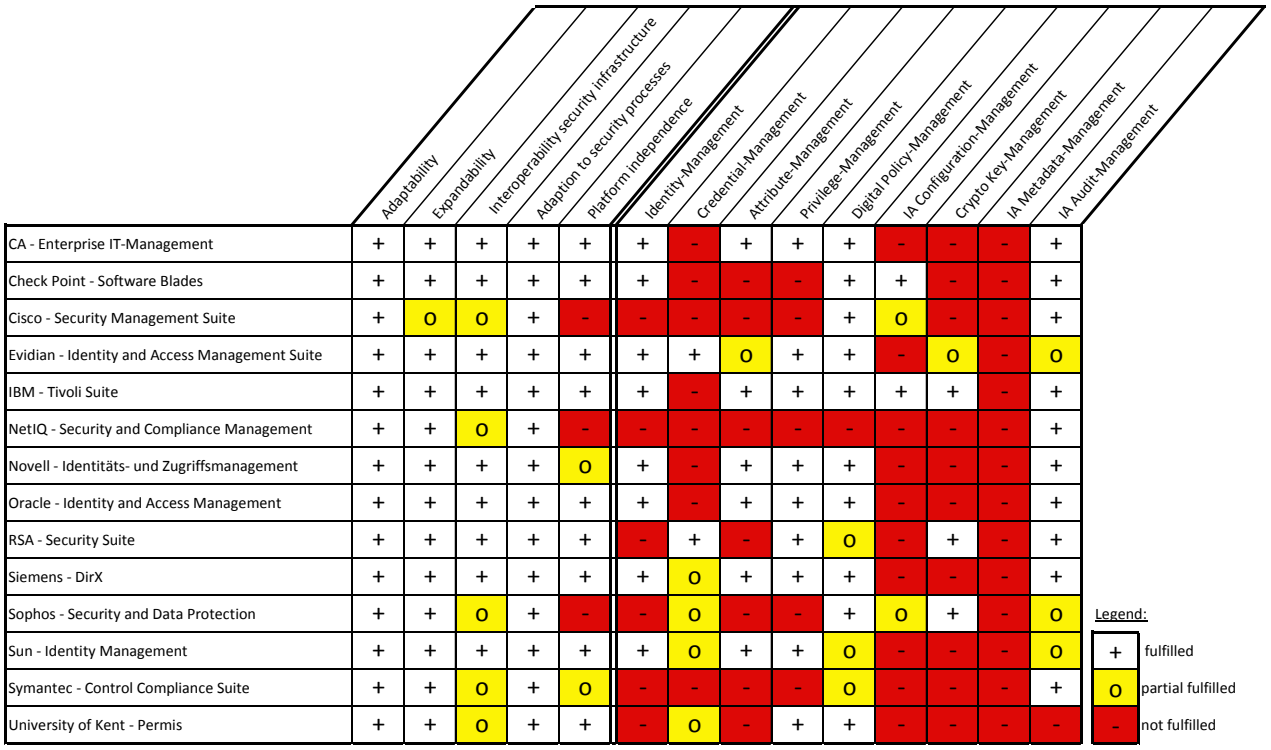


Figure 2. Analysis of current security management approaches

authentication credentials. The IdM agent API (if available) provides the access identity attributes or provided integration kits for CA SiteMinder, Oracle Access Manager (COREid) and Tivoli Access Manager. PingFederate allows a service provider enterprise to accept SAML assertions and provide single-sign-on to applications for Citrix, SharePoint and Salesforce.com [14], [20].

C. Security management objects within interfaces

Interfaces and APIs, for Cloud portability and interoperability, include management and security issues. For example, security in the context of Cloud Data Management Interface (CDMI) refers to the protective measures employed in managing and accessing data and storage. CDMI can be accessed by protocols like SAN, NAS, FTP, WebDAV or REST. Security management measures within CDMI can be summarized as user and entity authentication, authorization and access controls, data integrity, data at-rest encryption, crypto key management, audit and meta-data management [21]. Some of these security management attributes are *cdmi_security_audit* (If present and 'true', the cloud storage system supports audit logging), *cdmi_security_data_integrity* (If present and 'true', the cloud storage system supports data integrity/authenticity) or *cdmi_security_encryption* (If present and 'true', the cloud storage system supports data at-rest Encryption).

VI. Cloud Security Management Reference Model

In this section an reference model for Cloud security management is presented based on the detailed requirement analysis and the Cloud security managed objects. The CSMRM, which is shown in Figure 3, will serve as a comprehensive guideline in order to implement and design Cloud security management systems that address the highlighted security spectrum.

Within the requirement analysis we identified four domains in order to cover the range of requirements and functions for security management. Consequently the CSMRM consists of four interoperating layers - *Adapter and Libraries Layer, Platform Service Layer, Functional Service Layer, and Collaboration Service Layer*. The CSMRM adopts a standards-based system design and implementation that enable interoperability, facilitate federated security management operations, allow the use of commercial products and ease evolution. Therefore, it includes 5 domains of interfaces and APIs for the Collaborative Cloud environment. Here a Cloud security management system has to interact with (1) the Security Management Infrastructure of the organization, (2) the security managed objects, (3) other Cloud security

 ISBN: 978-1-61208-106-9

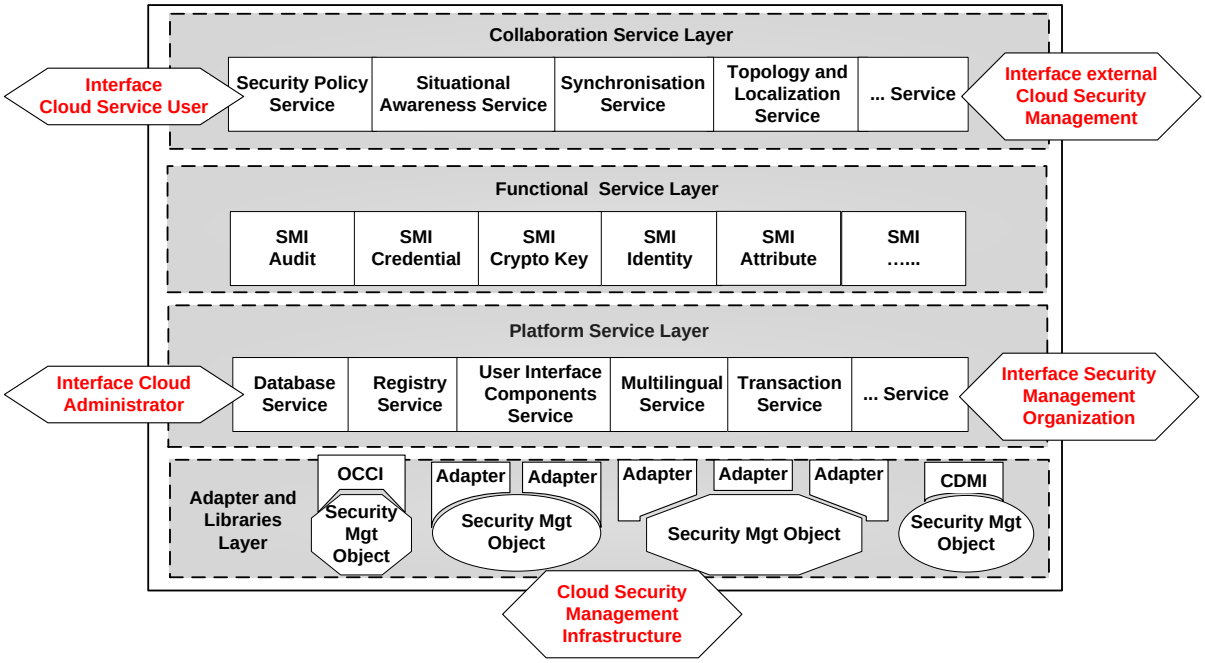


Figure 3. Cloud Security Management Reference Model

management systems (e.g., Cloud Broker, partners, public Cloud providers), (4) Cloud Service User, and (5) Cloud security roles within the organization (e.g., administrator, security officer, etc.). The lowest layer is called *Adapter and Libraries Layer*, which integrates and accesses various Cloud security management objects. These objects differ in two dimensions: First, how a the object is interfaced with. Each one may have its own protocol for communication (e.g., LDAP, proprietary APIs, or Simple Object Access Protocol (SOAP) in case of a Web Service). Second, the function a concrete object provides. The requirement to map all these different protocols and function sets to a Cloud security management system bears therefore a high degree of complexity. The proposed solution is abstraction and service decomposition, as the *Adapter and Libraries Layer* consists of different types of adapters and libraries, where each type may have a number of security management object-dependent implementations. Each adapter has two interfaces: a primitive function interface, which is common for all adapters of one type, and a object-dependent interface, which implements the (proprietary) interface of the concrete security managed objects. If available the adapter type will be defined and implemented according to standards like SAML, KMIP, OCCI, or CDMI. Above that the *Platform Service Layer* provides basic services to the Cloud security management system and implements the integration within the SMI of the organization. For example a *Database Service* including backup functionality will serve as the underlying security management data storage for the future system. For specific purposes different database types like structured and unstructured ones are offered. A *Registry Service* is necessary in order to have an overview about all system components within the Cloud security management system. Further a *Multi-lingual Service* allows a user to define, select, and change between different culturally-related application environments to support the usage within Collaborative Clouds. The *Functional Service Layer* includes all security management functional areas like Identity Management, Privilege Management, Metadata Management, Policy Management and Crypto Key Management. These services comprise all elements of their area within one organization. A *Collaboration Service Layer* is at the top of the CSMRM that support the inter-security management information, where Cloud security management applications of two or more organizations share information in Collaborative Clouds. The use of standardized and non-proprietary protocols to communicate and exchange information between security capabilities will support this inter-organizational sharing of information to prevent vendor lock-in threat. A *Security Policy Service* guarantees that regulations and constrains of the organization are enforced even when the combined Cloud service respective security data is distributed and located at various geographic units within the collaboration. In addition a *Topology and Localisation Service* provides an up-to-date view of the orchestrated collaborative environment, that supports other services of that layer. Underlying to these is a *Synchronization Service* that allow the exchange and transaction between various technologies and timezones.

 ISBN: 978-1-61208-106-9

VII. CONCLUSION

Identifying and defining security management issues for Cloud Computing are challenging tasks. Though inadequate security management in order to establish trust and preventing risks can be the show stopper for ubiquitous Cloud usage as Cloud services will multiply and expand faster than the ability of Cloud consumers to manage or govern them in use. Ubiquitous connectivity, the amorphous nature of information interchange, and the ineffectiveness of traditional static security controls which cannot deal with the dynamic nature of Cloud services, all require new security thinking with regard to Cloud Computing in the context of Collaborative Clouds. However an objective about how this new computing paradigm will impact organizations from a security management perspective, or how it can be used with existing technologies, and the potential pitfalls of proprietary technologies that can lead to lock-in and limited choice, is needed. To overcome this situation we presented a Cloud Security Management Reference Model that enables potential users to manage various horizontal and vertical Cloud security services independent of their complexity. The adapter and library layer allows the integration and clustering according SMI functions that guarantees a comprehensive coverage of all security management aspects. Furthermore it support the collaboration with Cloud service users and other Clouds. Due to the fact that such a comprehensive Cloud security management model does not exists yet, it will serve as a guideline to design and implement future Cloud security management systems.

ACKNOWLEDGEMENT

The authors wish to thank the members of the Chair for Communication Systems and Internet Services at the Universität der Bundeswehr Munich, headed by Prof. Dr. Gabi Dreo Rodosek, for helpful discussions and valuable comments on previous versions of this paper. The Chair is part of the Munich Network Management Team. Furthermore, we would like to acknowledge the support and contribution of the NATO SC/4 SMI AHWG and the European Defence Agency's PT CIS. This research activity has been performed partially in cooperation with the Federal Office for Information Security.

REFERENCES

[1] THE 451 GROUP, *CLOUDSCAPE - Cloud Codex*, www.451group.com/reports/executive_summary.php?id=869, last access: 23-08-20010

[2] Cloud Security Alliance, *Security Guidance for Critical Areas of Focus in Cloud Computing V2.1*, 2009

[3] R. Miller, *Cloud Brokers: The Next Big Opportunity?*, Data Center Knowledge, http://www.datacenterknowledge.com/archives/2009/07/27/cloud-brokers-the-next-big-opportunity/, last access: 23-08-20010

[4] A.V. Dastjerdi, K.A. Bakar, and S.G.H. Tabatabaei, *Distributed Intrusion Detection in Clouds Using Mobile Agents*, Advanced Engineering Computing and Applications in Sciences, 2009. ADVCOMP '09. Third International Conference on , pp.175-180, 2009

[5] EDA, *End-to-End Security Management in a Heterogeneous Environment*, EDA 08-CAP-027, 2009

[6] NATO, *Concept of a NATO Security Management Infrastructure*, AC/322(SC/4-AHWG/3)WP(2007)0001), 2008

[7] NSA, *Enterprise Security Management: A context overview*, 2009

[8] CESG, *Study on Security Management Infrastructure*, http://www.cesg.gov.uk/, last access: 23-08-20010

[9] M. Kretzschmar and F. Eyermann, *A Multi-Layer Architecture for a Security Management Infrastructure* , MilCIS 2009 Conference Paper, 2009

[10] Jericho Forum, *Cloud Cube Model v1.0: Selecting Cloud Formations for Secure Collaboration*, 2009

[11] DMTF - Open Cloud Standards Incubator, *Interoperable Clouds-A White Paper from the Open Cloud Standards Incubator*, 2009

[12] SIT Fraunhofer, *Cloud-Computing-Sicherheit*, http://www.sit.fraunhofer.de/pressedownloads/artikel/bestellung_ccs.jsp, last access: 23-08-20010

[13] B. Kandukuri, V. Paturi, and A. Rakshit, *Cloud Security Issues*, Services Computing Conference 2009 - SCC '09, pp. 517 - 520, 2009

[14] J. Rhoton, *Cloud Computing Explained: Implementation Handbook for Enterprises*, Recursive Press, 2010

[15] T. W. Wlodarczyk, C. Rong, and K. A. Haaland Thorsen,*Industrial Cloud: Toward Inter-enterprise Integration*, Cloud Computing, 2009

[16] C. Opincaru, *Service Oriented Security architecture applied to Spatial Data Infrastructures*, Universität der Bundeswehr München, Dissertation, 2008

[17] M. Knüpfer, *Analyse und Bewertung von SMI Anwendungen*, Bachelor thesis (in German), Information System Laboratory, University of the Federal Armed Forces Munich, Germany, 2010

[18] Y. Jung and M. Chung, *Adaptive security management model in the cloud computing environment*, Advanced Communication Technology (ICACT) 2010, vol. 2 pp. 1664 - 1669, 2010

[19] Amazon, *Amazon Web Services: Overview of Security Processes*, http://aws.amazon.com/de/security/, last access: 23-08-20010

[20] PingIdentity, *PingFederate*, http://www.pingidentity.com, last access: 23-08-20010

[21] SNIA, *Cloud Data Management Interface (CDMI) v1.0*, http://www.developersolutions.org/, last access: 23-08-20010

 ISBN: 978-1-61208-106-9

C2TP: A Service Model for Cloud

Chris Peiris
Faculty of Information Sciences and Engineering
University of Canberra
ACT, 2601, Australia
Chris.Peiris@canberra.edu.au

Dharmendra Sharma
Faculty of Information Sciences and Engineering
University of Canberra
ACT 2601, Australia
Dharmendra.Sharma@canberra.edu.au

Bala Balachandran
Faculty of Information Sciences and Engineering
University of Canberra
ACT 2601, Australia
Bala.Balachandran@canberra.edu.au

Abstract **- The notion of cloud computing capability is gathering momentum rapidly. However, the governance and enterprise architecture to obtain repeatable, scalable and secure business outcomes from cloud computing is still greatly undefined. There is very little research explored to define a framework that not only considers financial motivations, but also business initiatives, IT governance structures, IT operational control structures and technical architecture requirements to evaluate the benefits regarding cloud investment. We are proposing a novel model to address this. This model can be leveraged by an organization to evaluate the "tipping point" where the organization can make an informative decision to embrace cloud computing at the expense of on-premise hosting options. The authors refer to this model as Cloud Computing Tipping Point (C2TP) model. The model is a service centric framework created by mapping cloud computing attributes with industry best practices such as ValIT, Control Objectives for Information and related Technology (COBIT) and Information Technology Infrastructure Library (ITIL). This paper discusses the C2TP model in detail with its findings.**

Keywords - Cloud computing Tipping Point; C2TP; Cloud readiness model; CTPXML; Cloud artifact; Cloud taxonomy

I. INTRODUCTION

Merrill Lynch research estimated cloud computing as a "$160 billion addressable market opportunity, including $95 billion in business productivity applications, and another $65 billion in online advertising" [11]. There are several industry vendors (e.g., Amazon, Google, Microsoft, Sun, Salesforce, HP, etc.) attempting to capitalize on this market opportunity. Academic research community has also taken a keen interest in creating frameworks such as Virtual Workspaces [18], OpenNebula [22], Eucalyptus [10],[39] and Aneka [5] to address this opportunity. According to Buyya & Yeo [5], the hype about cloud computing is getting realized in the form of real world solutions. They continue to elaborate by focusing on computation power as the 5th utility on top of water, electricity, gas and telephony [6]. This computing utility will provide basic computation requirements for essential every day needs similar to the other four utilities stated above. They identify cloud computing is one of the paradigms that could realize this vision [6].

An ICT organization will reach a "cross road" or an "equilibrium" where the organization is required to make a conscious decision to either enhance the existing *on premise* investment or procure IT capability via cloud computing providers [26]. The popularity of cloud offerings are primarily driven by the financial benefits gained by organizations in comparison to *on premise* investments. Buyya, Pandey & Vecchiola [4] proposed a market oriented cloud computing architecture leveraging "Cloudbus" toolkit. The Cloudbus architecture addresses simulations, policies and algorithms to facilitate the cloud marketplace. Lenk et al. [20] created a general framework that targets transitioning from existing systems to cloud offerings. This framework primarily focuses on the financial advantages of the cloud platforms. Both of these capabilities do not address a model that an organization can leverage to evaluate organization's business initiatives, existing IT investment and existing IT control structures to determine the equilibrium (or a "tipping point") for cloud computing investment. This information is vital to evaluate the decision to either migrate to the cloud or expand on existing investment for an organization.

This paper attempts to address this research gap. The focus of this paper is to demonstrate Cloud Computing Tipping Point (C2TP) model that an ICT organization can leverage to evaluate the future benefits or limitations. The authors developed the following approach to create the C2TP model as detailed in Figure 1.

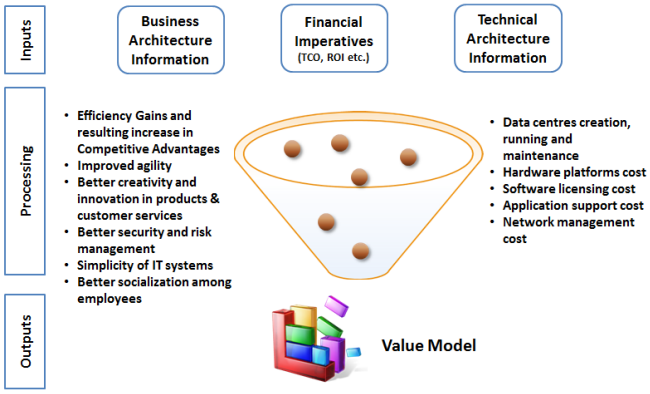


Figure 1. C2TP model inputs, processing and outputs

The authors have leveraged multiple case studies representing both enterprises and small to medium enterprises to investigate the relevant attributes for the perceived model. The model primarily focuses on organization's business architecture information, financial viability information and technical architecture information [27]. They are described as "inputs" of the model in Figure 1. The model captures many attributes that represent the above focus areas. This is illustrated as "processing" in Figure 1. These subject areas are discussed in detail in the next section. They are analyzed and evaluated to generate multiple measurements and services. Figure 1 illustrates these measurements and services as "outputs". These measurements and services are described in sections II and IV. These services can be leveraged to provide guidance regarding the "readiness" of the organization to embrace or decline on cloud offerings. Section III of this paper describes the details of an artifact designed to evaluate and capture findings of C2TP model. Section V will detail the findings, conclusions, future work and related work.

II. C2TP MODEL

The C2TP model will enable organizations with the knowledge to invest in future strategic decisions regarding cloud computing investment. The model will evaluate financial, business and technical data to derive a conclusion whether the organization will benefit from investing in cloud computing or extend their investment in *on premise* capability. This is an important toolset that can be leveraged by the industry and will actively contribute to cloud computing enterprise architecture. The following Figure 2 illustrates components of C2TP model.

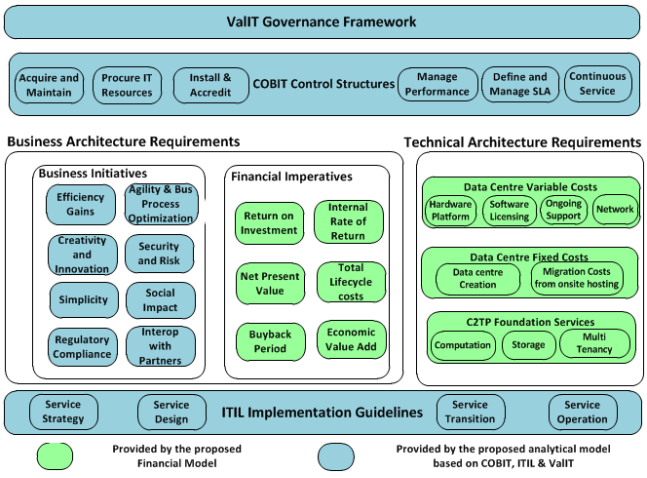


Figure 2. C2TP : Cloud computing Tipping Point Model

The authors explored whether they could use industry best practices of Control Objectives for Information and related Technology (COBIT) [7] and Information Technology Infrastructure Library (ITIL) [14] as a base framework to define the attributes for the Cloud Computing Tipping Point model. Lainhart [19] described COBIT as a "methodology for managing and controlling Information. It also controls Information Technology risks and vulnerabilities". IT Infrastructure Library (ITIL) is a set of industry best practices published by British Office of Government Commerce (OGC) [14]. It comprises of operational framework details that includes industry best practices, standard operating procedures and technical operating procedures. The authors concluded that the high usage of these frameworks in the industry and their proven control structures provide a solid foundation for the C2TP model. The primary attributes planned for the model are,

a) Better risk management and information security
b) Better software development life cycle management
c) Better business continuity
d) Excellent capacity and availability management
e) Low cost operations
f) Shorter implementation life cycles of IT systems and applications
g) Enhanced service levels
h) Higher uptime

The authors also concluded that some of the long term strategic objectives (e.g., value management, optimizing internal rate of return etc.) of a Cloud solution are not addressed by COBIT and ITIL control structures [29]. Therefore, they have leveraged ValIT [36] model to address these gaps. ValIT addresses assumptions, costs, risks and outcomes related to a balanced portfolio of IT-enabled business investments. It also provides benchmarking capability and allows enterprises to exchange experiences on best practices for value management [36]. Therefore, the authors have decided to leverage COBIT, ITIL and ValIT to provide the governance framework for Cloud Computing Tipping Point model. The authors have created key metrics to measure the outcome of the model [28]. They are,

- C2TP Financial Model metric.
- C2TP Business Initiatives Index.
- C2TP Operational Governance Index.
- C2TP Readiness Index.

A. C2TP Financial Model

The C2TP financial model analyses the fixed and variables costs of migrating and sustaining cloud platform architecture. The financial model currently accommodates the following costs. These are staff costs, help desk costs, server maintenance costs, server replacement costs, data center environment costs, software licensing costs, software maintenance costs, networking costs, storage costs, migration costs, centralization costs, journaling costs, training costs, archiving infrastructure costs, archiving content costs, backup and recovery costs, disaster recovery costs, support consultancy costs and cost of Investment (interest on borrowed funds to facilitate the investment). Future work on this base framework can expand these criteria further.

 ISBN: 978-1-61208-106-9

The model analyses the above costs for *on premise* IT investment and will compare it against the cloud alternative capability. The model evaluates the financial position of the *on premise* investment and cloud offerings by referring the following equation. Note that "n" is a positive integer and "N" is the total number of financial costs.

$$\sum_{n=0}^{N}(Cost_{Cloud}) \; < \; \sum_{n=0}^{N}(Cost_{On\ Premise})$$

An organization is financially in a better position if the cloud offerings investment is less than the *on premise* investment.

B. C2TP Business Initiatives Index

C2TP business initiatives index evaluates the organization's eagerness and ability to embrace cloud computing as a mechanism to gain completive advantage over their peers. The business initiative attributes of the model are,

- Efficiency gains
- Agility
- Creativity and Innovation
- IT Security issues
- Risk Management
- Simplicity of capability development and management
- Business process optimization
- Social impact to the employees
- Regulatory compliance
- Interoperability with partner organizations.

The C2TP model gathers the organization's perceptions and priorities regarding the above subject areas. This information is gathered by answering series of questions to evaluate the organization's perspective of these business initiatives under the *on premise* model and alternative cloud computing capability. These business questions leverage ValIT, COBIT and ITIL key subject areas. The total of *on premise* business initiatives are collected together as "On Premise Business Initiatives index". The total of cloud business initiatives are collected together as "Cloud Business Initiatives Index". In order to move to the cloud, the Cloud Business Initiatives Index requires to be greater than the On premise Business Initiatives index as described below. Note that "n" is a positive integer and "N" is the total number of business initiative measures. (Note – Business Initiatives Index is abbreviated as "BII")

$$BII_{\ Cloud} \; = \sum_{n=0}^{N}(Business\ measure_{\ Cloud})$$

$$BII_{\ On\ Premise} \; = \sum_{n=0}^{N}(Business\ measure_{\ On\ Premise})$$

$$BII_{\ Cloud} \quad > \quad BII_{\ On\ Premise}$$

C. C2TP Operational governance Index.

This index attempts to compare the organization IT governance and control objectives under the *on premise* and cloud computing models. This measure is vital to ensure the organizations future IT platform (regardless of *on premise* or cloud computing) is operating as efficiently as possible. COBIT and ITIL provides extensive capability in industry implementations for similar requirements. The authors have evaluated these industry best practices extensively. Therefore, the C2TP model is influenced by COBIT and ITIL control structures to address some of the key capabilities as described below.

1) Acquire and Maintain Application Software

The key focus areas under this model are,

- Which model is more efficient in procuring software?
- Which model attracts the best licensing costs arrangement for the organization?
- Which model manages the licensing costs better? (e.g., better tools are available for management and transparency)
- Which model provides more efficient application maintenance?
- Under which model is application maintenance financially attractive?

2) Procure IT Resources

The key focus areas under this model are,

- Which model provides the procurement efficiencies? (i.e., less time to procure software)
- Under which model is it easier to secure IT resources that are skilled to develop IT capability?
- Which model offers cheaper to find IT resources?
- Which model offers efficient project management?

3) Acquire and Maintain Technology Infrastructure.

The key focus areas under this model are,

- Which model offers simpler procurement process to acquire hardware resources?
- Which model offers lower infrastructure costs for the organization?
- Which model offers better hardware infrastructure optimization?
- Which model offers better tools to address disaster recovery functions?
- Which offers better toolset to manage the organization's infrastructure?

4) Manage Performance and Capacity

The key focus areas under this model are,

- Which model offers better performance tools set to monitor hosted applications?
- Which model offers better performance tools set to manage hosted applications?
- Which model provides easier access to increase or decrease capacity?

5) Define and Manage Service Level

The key focus areas under this model are,

- Which model offers better Service Level Agreements (SLA) for organization needs?
- Which model offers liability measures (from providers) in case of lack or interruption of service or data?
- Which model offers clear 'pay per view' cost structure for services consumed?

 ISBN: 978-1-61208-106-9

6) Ensure Continuous Service

The key focus areas under this model are,

- Which model offers better availability of data?
- Which model offers better availability of critical services?
- Which model ensures integrity of data?
- Which model ensures integrity of critical services?
- Which model ensures confidentiality of corporate data?
- Which model secures organization from Denial of Service attacks?

C2TP model evaluates these subject areas by engaging with the target organization with a series of workshops, questions and answers sessions and presentations. The interactions with the organizations and the organization's responses are recorded in the C2TP artifact tool. This tool and its functionality will be described in section III. The organization evaluates its *on premise* business case against the cloud computing alternative under the C2TP model. The C2TP model proceeds to accumulate this information and derives Operational Governance Index for *on premise* and cloud computing alternatives. Then the model compares the two indexes to analyze the suitability of each model. The Operational Governance Index of cloud computing should be greater than the Operation Governance Index of *on premise* model in order for an organization to benefit from cloud computing offerings. The equations to obtain this decision are demonstrated below. Note that "*n*" is a positive integer and "*N*" is the total number of operational governance measures. (Note - Operational Governance Index is abbreviated as "OGI".)

$$OGI_{Cloud} = \sum_{n=0}^{N} (Operational\ Gov.\ measure_{Cloud})$$

$$OGI_{On\ Premise} = \sum_{n=0}^{N} (Operational\ Gov.\ measure_{On\ Premise})$$

$$OGI_{Cloud} > OGI_{On\ Premise}$$

D. C2TP Readiness Index.

C2TP model readiness index is the end result of the evaluation. The readiness index will elaborate the organization's "readiness" to embrace cloud computing or whether it has reached the "tipping point" to invest in cloud computing offerings. The Readiness Index is derived by aggregating the previous discussed indexes. The authors believes the C2TP readiness index and all the other indexes needs to be positive to conclusively decide that the organizations have reached the "tipping point" to invest in cloud computing offerings. The following illustrates this equation.

C2TP Readiness Index =
Financial Model + Business Initiatives Index + Operational Governance Index

The authors developed an artifact to demonstrate this model and to evaluate its capability extensively. This conclusion of C2TP Readiness Index was confirmed by the findings of the artifact.

III. C2TP ARTIFACT

This section discusses the Cloud Computing Tipping Point artifact that was designed and developed to evaluate and captures the information in relation to C2TP model.

A. Research method – Artifact building

Artifact design is an iterative process. March and Smith believes that the search for the best, or optimal, design is often intractable for realistic information systems problems [23]. Simon [32] describes the nature of the design process as a Generate and Test Cycle. According to Simon, we would generate research material and will evolve to new material when we apply testing scenarios. The authors have followed this process numerous times to create the artifact for C2TP model. March and Smith [23] proposed 4 general outputs for design research. They are constructs, models, methods, and instantiations. The C2TP model addresses all 4 of these outputs.

- Constructs are defined by March and Smith [23] as conceptual vocabulary of problems and solution domains. This is addressed by defining extensive conceptual architecture of the C2TP artifact. It breaks down the conceptual architecture to multiple layers (i.e., presentation, business logic and database) and clearly defines each layer's responsibility.
- A model is a set of propositions or statements expressing relationship among constructs [23]. The artifact addresses this requirement by extensive set of Unified Mark-up Language (UML) constructs. They are outputs such as entity relationship diagrams and class diagrams.
- The method is a set of steps used to perform a task [23]. The artifact leverages UML modeling techniques such as use cases and sequence diagrams to address this.
- March and Smith [23] defines instantiations as "operationalize constructs, models and methods. The eventual instantiation of the research is a web site that accumulates all the information regarding the C2TP model.

B. Solution Architecture for the model

The C2TP conceptual architecture is based on three tier architecture model. The three tiers are Presentation, Business Logic and Database. There are also several "cross cutting" components of the architecture including security, common integration methods and a library of components that addresses instrumentation, auditing, logging, tracing, caching and validation. The authors have created a specific taxonomy to share C2TP model information. This taxonomy is referred to as Cloud Tipping Point Markup Language (CTPXML) [29]. CTPXML is a well formed communication structure specifically designed for C2TP model. The primary objective of CTPXML is to exchange information between C2TP model and its consumers in a scalable and secure environment. The consumers can be number of entities. They

 ISBN: 978-1-61208-106-9

are web browsers, mobile phones, desktop applications or intelligent agents [30].

Presentation layer essentially consists of user experience layer and self-service channel to expose the C2TP model capability. The self-service channel will provide the ability to expose a "service view" of C2TP selected applications to be consumed by external service providers and 3rd parties. The Self Service Channel will host scalable and reliable services that will allow other organizations to build their own applications that leverage C2TP data. However, the application creation and maintenance will be the sole responsibility of the 3rd party organization. The following services will be exposed via these mechanisms that are built on top of the "service agents" that technology platform provides [30].

- C2TP Financial Model Service
- C2TP Business Initiatives Index service
- C2TP Operational Governance Index service
- C2TP Readiness Index service

C2TP Business Logic layer accommodates both in-host and cross-host capabilities. The in-process model supports the web user, mobile user and potential desktop forms application functionality. The in-process model enables memory caching and will provide speedy access to data. The services interface provides a scalable standard interfaces for the Self Service Channel. This layer will provide the proxies or agents that facilitate the plumbing code for the service layer. It also leverages standard service contracts, message contracts, data contracts and fault contracts to communicate to presentation layer. The orchestration layer of the business logic layer is a key capability. The orchestration layer will route client requests to the relevant business components. The orchestration layer coordinates (and in some cases aggregates) data communications between business components and necessary integration components. This layer will also provide framework level support for the following,

- Validation capability – these components provide business level validations (e.g., valid post code) and developer support (e.g., regular expressions to validate email address)
- Business rules and business objects – The business logic and the associated business rules are hosted in these objects.
- The necessary workflow rules and components.
- Client message inspector – This component evaluates the user's authorization credentials to execute the business functionality.
- Parameter inspector – This component will inspect the parameters being provided to the business layer by the presentation layer.

The database solution architecture supports necessary table structures and integration connectors. The database design will follow the 3rd normal form and there are small data marts created for user and administrator reports.

C. Service enablement of C2TP model

SOA is a software architecture that is designed around loosely coupled software components called services, which can be orchestrated to improve business agility [8]. Vouk [37] defines SOA as delivery of an integrated and orchestrated suite of functions to an end-user through composition of both loosely and tightly coupled functions, or often networkbased services. As defined by W3C, services provide functionality at the application and business levels of granularity using widely applied standards [13] . A conceptual SOA metamodel to enable business capability was demostared by Emig et al [9] and Henkal & Zdravkovic [12]. This SOA metamodel is leveraged to build the C2TP artifact that can seemlessly intercat with 3rd party compoments or intelligent agents to share information generated by the C2TP model.

IV. RESULTS AND FINDINGS

The proposed model has been validated with results from several organizations with a cross-section of business models. Due to the recent global financial crisis and volatility of the global markets, the organizations were quite keen to investigate technology options that give them a competitor advantage over their competitors. Therefore, C2TP model was viewed as a timely development to assist with their evaluation for cloud computing options.

'One on one' interviews and workshops were the primary mechanisms of collecting data and evaluating C2TP model with target organizations. The collected information was entered into the C2TP artifact to process the results. The C2TP artifact produced the financial model and the relevant indexes to evaluate the cloud computing suitability. The results were member checked [21] with the participants. Member checking was performed both during the interview (or workshop) process and at the conclusion phase. Interviewer corroboration [21] was also leveraged as a validation technique to ensure the quality of results. Negative case analysis [34] provided a valuable toolset to refine the model further in some cases. The paper categorized these findings under two categories to address necessary cross sections of the IT industry. They are as enterprise clients and small or medium enterprise clients.

A. Validation of Enterprise clients

Experiment 1 - C2TP model was evaluated by representatives of a global consulting company that has in excess of 80,000 employees. The organization is a mature CMMI level 5 accredited IT services provider. The representatives' motivations were to analyze the cloud email hosting benefits of transferring their *on premise* capability to cloud platform. They were interesting in leveraging both Infrastructure as a service and software as a service cloud offerings. They leveraged the C2TP model to comprehensively analyze their requirements. Here are the results.

- The model clearly identified the financial benefits leveraging the formulae discussed earlier. The C2TP Financial readiness Index was positive.

- The C2TP business Initiatives index was also positive. This supported the conclusion of the organization business initiatives are in synergy with the cloud platform offerings.
- C2TP Operational Governance Index was positive. The organization's accreditation of CMMI level 5 had ensured structured processes to address management and governance issues. Therefore, the organization could effectively transition their *on premise* email hosting environment to cloud computing offering with minimum number of business interruptions.
- The C2TP model concluded that the C2TP Readiness Index was positive in light of all the other indexes were positive. Therefore, the C2TP model endorsed the organizations strategy to migrate their email hosting system to cloud platform and start leveraging the benefits of the cloud architecture.

The participants provided valuable feedback as part of member checking to enhance the C2TP model. They have indicated the value of benchmarking their findings with similar organizations to further understand the competitor advantages and limitations. They have highlighted that a rich set of historical data that focuses on their industry does add substantial value to the existing C2TP model. This important discovery was corroborated by other interviewers subsequently.

Experiment 2 - The 2nd enterprise company is a global supplier of security hardware to financial industry and government agencies. They have close to 20,000 global employees. The company representatives' interest was to leverage the C2TP model to evaluate their "Next generation desktop" project. This is a platform rationalization project viewed as an opportunity to provide a solution for their existing aging desktop platform. This platform as a service project also attempts to provide a single environment for the company's newly acquired subsidiaries to work together and to be integrated into their global headquarters. Here are the findings

- The C2TP Financial Readiness Index was positive indicating the financial benefits of rationalizing their platform with cloud platform offerings.
- The C2TP Business Initiatives Index was negative. The company subsidiaries operated as independent entities in all its geographies. Therefore, their business initiatives were influenced heavily by demographic circumstances and were not centrally managed. This was the outcome of several acquisitions and mergers over the years. Unfortunately, the IT environment integration wasn't a priority in these acquisitions and mergers. The objective of the "Next generation desktop" was to provide a unified desktop solution globally. However, the representatives could not agree on the unified set of business initiatives that was driving the project.
- C2TP Operational Governance Index was negative. Due to mergers and acquisitions, there were substantial differences between the IT maturities for IT teams representing different geographies. This led to the conclusion that they need to create a unified IT platform and a governance framework at the global headquarters level before they investigate migrating capability to the cloud. It was noted that this global platform could be viewed as a Cloud offering. However, the representative did conclude there are sensitive data (as a result of their security operations) that they were not planning to host on the cloud offering. They concluded to consolidate all the IT teams and create governance framework. This governance framework will address the issue of sensitive data and investigate leveraging cloud computing for their non-core activities.
- Due to the C2TP Business Initiatives index and C2TP Operational governance index being negative, it was concluded that the C2TP Readiness Index was negative. Therefore, it was concluded that the organization haven't reached the "tipping point" to invest in cloud offerings.
- The conclusion was to enhance their existing invest in on premise offering and enhance the maturity of the organization till it reaches a point where they can revisit the cloud computing value proposition.

The paper leveraged negative case analysis validation technique to refine the model with this particular experiment. The initial data from the C2TP model did not corroborate with the interview findings. Further analysis led to the revision of the model attributes and introducing a "weightings" system for the calculations to reflect these special circumstances.

B. Validation of SME clients

Experiment 3 - The model was evaluated by an Australian consulting company that specializes in collaboration and portal development capability. This company has close to 20 employees. They specialize in collaboration toolset for financial and manufacturing sector. The company was having difficulty to scale their current *on premise* infrastructure platform to facilitate its growing demand. They also noticed high seasonal demand for Collaboration capability towards the start of the financial year by their financial industry clients. The company was having difficulty to justify the additional costs to cater small window during seasonal spikes and leave the resource idle for rest of the year. They were keen to leverage the C2TP model to explore their options and evaluate their readiness to acquire cloud computing capability. Here are the results,

- Positive C2TP Financial Model for the company elaborating the advantages of moving to a cloud provider to facilitate the seasonal spikes in lieu of acquiring new *on premise* infrastructure.
- Due to the small nature of the company and mature ITIL process in place, the C2TP Operational Governance Index was positive.
- Their business Initiatives index was also positive indicating their suitability to migrate to the cloud platform.

 ISBN: 978-1-61208-106-9

- The C2TP readiness index was positive due to the feedback from C2TP Financial Model, Business Initiatives Index and Operational Governance Index

Member checking and interviewer corroboration were leveraged as validation techniques for this experiment. The organization was keen to leverage C2TP model to investigate sharing CTPXML information with their partner organizations and clients.

Experiment 4 - The 2nd SME participant organization specializes in residential and commercial imagery for product catalogues. This is a small organization with 5 employees. The major challenge was to manage their digital library and keep up to date with technology advancements and security considerations. They were interested in evaluating both platform as a service and software as a service offering to meet their growing demand. They have been leveraging their *on premise* platform for number of years. However, due to increased storage needs, image processing and bandwidth needs, they were finding it difficult to extend their existing infrastructure and justify the extra costs. Therefore, they were keen to evaluate the C2TP model to obtain guidance to make decisions on investing on cloud offerings

- The C2TP Financial Model was positive indicating they were financially in a better position due to their move to the cloud offerings.
- The C2TP business Initiatives Index was also positive indicating that the business models of the company or the business initiatives are not adversely affected by transferring their capabilities to the cloud. They also noticed that cloud offering will significantly reduce their capital expenditure and transfer costs to the "operating budget".
- C2TP Operational Governance Index was positive. The company benefitted from its small size and was able to be agile and improve its process. Therefore, they had sufficient control structures and governance model to successfully migrate to the cloud platform with little process alterations.
- Due to positive feedback from C2TP Financial Model, C2TP Business Initiatives Index and C2TP Operational Governance index, the C2TP Readiness Index was positive. The conclusion was that the organization has the capability and the motivations to successfully migrate to could platform under the C2TP evaluation.

These 4 experiments have addressed both enterprise and small to enterprise organizations to obtain a good cross section of the IT industry. The experiments also addressed variations of size, number of employees, industry focus and multiple geographical locations. C2TP model was leveraged by all 4 experiments successfully with proven validation techniques such as member checking, interview corroboration and negative case analysis. Therefore, the C2TP model has assisted in our research goal to create a model that provides strategic guidance to organizations to evaluate future cloud investments in comparison with *on premise* investment.

V. CONCLUSION AND FUTURE WORK

A. Summary of Contributions

The primary contribution of the research is the Cloud Computing Tipping Point (C2TP) model and the artifact to demonstrate its capability. This model will assist organizations to evaluate the "tipping point" whether the organization can either embrace the cloud offerings or enhance their investment in *on premise* IT capability. This model provides a comprehensive structure by leveraging proven financial indicators and industry best practices (i.e., COBIT, ITIL and ValIT). This model addresses a timely need in the industry to create a framework to provide guidance to organizations to evaluate the benefits and limitations of embracing the cloud platform. The current literature and the industry momentum are driven purely by the financial comparisons of *on premise* cost versus cloud computing costs. There is no model available that analyses not only the financials, but also the organization core business initiatives, its existing IT capability control structures and the suitability of the organizations existing IT governance processes in relation to cloud computing. These selection criteria have the same weight as the financial information for organizations

These organizations are aware of the complexities of IT projects. The complexity of these IT projects will further enhance if they include new cloud capability that the organization cannot manage or govern in a scalable and predictable fashion. These organizations also need to mitigate the risk of an external cloud provider having access to their business information. Does the cloud platform offer better "value for money" under these circumstances? Or does the organization conclude the organization's intellectual property information is too sensitive to share on the cloud? Unfortunately, there is no model available to address these concerns currently. The authors believe in a set of minimum criteria or checklist items for organizations to evaluate to determine their cloud readiness. Therefore, the authors believe the C2TP model provides a comprehensive evaluation criterion to address this growing need in the industry. The model provides feedback to an organization on its cloud readiness by factoring in the financials and other key business factors. Therefore, this model enhances the academic body of knowledge in relation to cloud enterprise architecture.

The cloud infrastructure is primarily built using virtualization technologies comprising of hypervisors running on a large number of parallel servers operating in distributed networking environment. The major hosting providers have also launched application programming interfaces for development of cloud friendly programs that can be run in Software as a Service mode. The current hosting providers have kept the backend architectures proprietary and hence there is a risk of the applications getting tied to specific cloud computing environments. The current researchers feel a need for standard cloud infrastructure and an enterprise model that all hosting providers should follow in order to ensure openness of the system from customer perspective. The authors share these

 ISBN: 978-1-61208-106-9

sentiments and shall contribute to such an open architecture model supporting the empirical generalizations being attempted by the current researchers. The C2TP model will contribute and enhance enterprise architecture of cloud computing as a result of this research.

This paper also analyses the synergy between the industry best practices (i.e., COBIT, ITIL and ValIT) and how they can be coupled together to create an academic model. This academic model then is tested and proven to be effective by referring to the previous section. Selected components of these industry best practices are indirectly peer reviewed and analyzed in academic context. This enhances the 'body of knowledge' of academic empirical studies based on these industry best practices. This will enhance credibility and wider acceptance in academia for these industry best practices for academics to be leveraged in the future.

The CTPXML taxonomy is another contributor as a result of this research. The CTPXML taxonomy is primarily designed as the communication mechanism to share data between C2TP model and its consumers. The taxonomy is based on well-formed XML standards and will follow Web Service Basic Profile standards to ensure interoperability between heterogeneous platforms. The current taxonomy addresses the C2TP model information, security credentials, infrastructure information and transport information. The security, transport and infrastructure information is kept to minimum due to the focus of the C2TP model information. The authors believe these areas will be a prime candidate for future work. The CTPXML taxonomy can be leveraged by multiple clients to render the C2TP information seamlessly to their preferred presentation medium.

B. Future Work

C2TP model has undergone several iterations already. The model is also evaluated with enterprise and SME organizations as we discussed earlier. The findings of these evaluations were very positive. The findings also shed some light regarding the further enhancements to the model that can be classified as future work.

1) Expanding the initial design of the C2TP artifact

Currently the weighting of the attributes in Business Initiatives Index and Organizational Governance Index are equal. Therefore, each attribute is weighted evenly. The SME case study on enterprise global manufacturing company and similar ones have indicated that this position can be enhanced with a specialized weighting system in the future. This is to reflect that some organizations work in specialized industries with specialized business motivations. Hence they are bias towards certain attributes. For an example, a firm that specializes in Defense IT security has greater dependency on security and risk management attributes. Therefore, enhanced focus should be allocated to some attributes to reflect these industry specific circumstances. Significant industry benchmarking needs to be conducted initially to revise the weighing structure.

Bateman and Wood [2] argues that cloud computing promotes "Green IT" as long as location of hardware and storage premises leverage renewable energy. According to Bajgoric [1] and Vykiukal, Wolf & Beck [38], cloud computing indirectly reduces greenhouse gases and CO2 emissions. Issa, Chang and Issa [16] have proposed a PESTEL (i.e., Political, Economic, Social, Technological, Environmental and Legal) evaluation regarding cloud computing sustainability. The authors believe future expansions of the C2TP model should evaluate these current research developments.

2) A multi agent design & implementation architecture

The CPXML Taxonomy and service design of C2TP model can be leveraged by intelligent multi agents to capture process and evaluate information. This can be achieved by multiple intelligent agents working in synergy by communicating with other intelligent agents adhering to specific rule set. The agents can control the flow of information and manage the constant communications between the nodes. The agents can also be mobilized to address different components of the model and use complex algorithms to analyze developing situation in light of variable user input. The intelligent agents will be able adapt to information provided by the user and propose different metrics to evaluate the cloud suitability under these circumstances. Here are some of the opportunities leveraging current research

- Son and Sim [33] are creating a multi-issue negotiation mechanism for cloud service reservations. These agents will be able to negotiate price and time slot among cloud partners. The C2TP agents will be able to communicate with these intelligent agents to conduct and report on these negotiations.
- Kang and Sim [17] are creating a cloud ontology and agent based cloud search engine referred to as "Cloudle". It is specifically designed to find cloud services over the internet. C2TP agents will be able to provide valuable information to expand this ontology and benefit form Cloudle search engine to locate complimentary cloud services.

3) Industry benchmarking for C2TP model

As discussed earlier, industry benchmarking has been identified as a key focus area for future work. It is expected that the C2TP model will evolve and refine according to the industry benchmarking feedback. Benchmarking and wide acceptance of the model will assist to refine the weighting system for attributes of the model. The C2TP model will evolve to comprise core competencies and will accumulate industry specific competencies as the result of the benchmarking activities. The participant organization will obtain richer and more competitive data analysis in their C2TP evaluation. Lenk and Nimis et al [20] has introduced an open framework to migrate the pre-existing cluster and grid computing capability to cloud computing. C2TP can leverage this framework to provide a future roadmap with the benchmark information as an implementation guide. This will provide both strategic and tactical information to the C2TP model consumer.

 ISBN: 978-1-61208-106-9

4) Enhancement of CTPXML taxonomy

The CPXML Taxonomy and service design of C2TP model can be leveraged by intelligent multi agents to capture, process and evaluate information. This can be achieved by multiple intelligent agents working in synergy by communicating with other intelligent agents adhering to specific rule set. An open implementation model has been proposed based on standard SOAP, UDDI and WSDL protocols. These standard protocols can be leveraged by intelligent multi agents to manage the C2TP evaluation process. The agents can control the flow of information and manage the constant communications between the nodes. The agents can also be mobilized to addresses different components of the model and use complex algorithms to analyze developing situation in light of variable user input. The intelligent agents will be able adapt to information provided by the user and propose different metrics to evaluate the cloud suitability under these circumstances.

The CTPXML taxonomy is expected to evolve with each new revision of the model. The benchmarking and refining of the model will introduce new taxonomy changes that need to be reflected in the communication with the model consumers through intelligent agents. It is also expected that the current model's security and transport capabilities will be enhanced. This will also result in some adaptation of the CTPXML taxonomy.

C. Related Work

Cloud computing popularity has prompted several academic and industry initiatives to explore the capabilities and enhancements in cloud computing. The value proposition of cloud computing in comparison with *on premise* investments is one of the key research areas. There are several initiatives to specifically address the economic viability of the cloud investment. They are primarily driven by industry vendors to promote their offerings. Cloud ROI Framework [15] and Azure ROI calculator [24] are two examples of this initiative.

There are several academic initiatives investigating key business model aspects of cloud computing. Buyya, Pandey & Vecchiola [4] proposed a market oriented architecture for cloud computing. This is a continuation of Buyya and Yeo et al. [6] attempt to introduce cloud economic model. Lenk et al [20] created a general framework that targets transition of existing systems to cloud platforms. Sun et al [35] discussed a SLA based model to facilitate financial services infrastructure. Brebner and Liu [3] compared various vendor offerings such as Google App Engine, Amazon EC2, and Microsoft Azure to provide guidance on cost, application performance (and limitations) for different deployment scenarios.

These academic initiatives are parallel related work to C2TP. However, all these initiatives are focused on financial benefits an organization can derive from cloud computing. Therefore, the C2TP model offers a unique research value by including organization's business initiatives, governance processes, existing control structures and technical architecture attributes on top of the financial imperatives. The paper believes the C2TP model contributes to the cloud computing enterprise architecture and will be leveraged by future academic research endeavors.

REFERENCES

[1] Bajgoric N. (2010), "Always-on Enterprise Information Systems for Business Continuance: Technologies for Reliable and Scalable Operations", IGI Global, USA

[2] Bateman A. and Wood M. (2009), "Cloud Computing", Bioinformatics, vol. 25, no. 12, p. 1475.

[3] Brebner P. and Liu, A. (2010), "Modeling Cloud Cost and Performance", Proceedings of 1st Annual International Conference on Cloud Computing and Virtualization, pp. 79-86, Singapore.

[4] Buyya R., Pandey S., and Vecchiola, C. (2009), "Cloudbus Toolkit for Market-Oriented Cloud Computing", Proceeding of the 1st International Conference on Cloud Computing (CloudCom 2009, Springer, Germany), Beijing, China.

[5] Buyya R., et al. (2009), "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility", Future Generation Computer Systems, Elsevier and Science Direct. Vol. 25: 599-616.

[6] Buyya R, Yeo C.S., and Venugopal S. (2008), "Market-oriented cloud computing: Vision, hype, and reality for delivering IT services as computing utilities", Proceeding of the 10th IEEE Int. Conference on High Performance Computing and Communications, HPCC 2008, Dalian, China, Sept. 2008.

[7] COBIT 4.1, (2007), – Executive Summary Framework. IT Governance Institute. ISACA.org. pp. 4-16.

[8] Erl T. (2005), Service-Oriented Architecture: concepts, technology and design, Prentice Hall, Upper Saddle River, NJ

[9] Emig C., Krutz K., Link S., Momm C., and Abeck S (2007), "Model-Driven Development of SOA Services", Cooperation & Management, Universität Karlsruhe (TH), Germany, p. 2.

[10] Eucalyptus Public Cloud (EPC). http://eucalyptus.cs.ucsb.edu/wiki/EucalyptusPublicCloud/, [Retrived on 5 May 2010]

[11] Hamilton D. (2008), "Cloud computing seen as next wave for technology investors". Financial Post, 04 June 2008, http://www.financialpost.com/money/story.html?id=562877

[12] Henkel M. and Zdravkovic J. (2005), "Approaches to Service Interface Design", Proceedings of the Web Service Interoperability Workshop, First International Conference on Interoperability of Enterprise Software and Applications (INTEROP-ESA'2005), Hermes Science Publisher, Geneva, Switzerland

[13] Huhns M. and Singh M. (2005), "IEEE Internet Computing, Published by the IEEE Computer Society, January edition.

[14] ITSMF (2001), "IT Service Management Version 2.1a", The IT Infrastructure Library, Office of Government Commerce UK.

[15] Infosys (2010), Cloud ROI Framework, http://www.infosysblogs.com/cloudcomputing/2009/06/the_cloud_roi_framework.html, [Retrieved on 27 March 2010]

[16] Issa T. and Chang V. (2010), "The Impact of Cloud Computing and Organizational Sustainability", Annual International Conference on Cloud Computing and Virtualization, pp. 163 – 169, Singapore

[17] Kang J. and Sim K. M. (2010), "An agent based Cloud Search Engine tat Consults a Cloud Ontology", Annual International Conference on Cloud Computing and Virtualization, pp. 312 – 318, Singapore

[18] Keahey K., Foster I., Freeman T., and Zhang X. (2005), "Virtual workspaces: Achieving quality of service and quality of life in the Grid", Scientific Programming, 13(4):265-275, October 2005

[19] Lainhart J.W. (2000), "COBIT: A Methodology for Managing and Controlling Information and Information Technology Risks and Vulnerabilities", Journal of Information Systems, Vol 14 (s-1), p. 21, doi: 10.2308/jis.2000.14.s-1.21

 ISBN: 978-1-61208-106-9

[20] Lenk A., Nimis J., Sandholm T., and Tai S. (2010), "A Multi -issues Negotiation Mechanism for Cloud Service Reservation", Annual International Conference on Cloud Computing and Virtualization, pp. 297 – 305, Singapore

[21] Lincoln Y. and Guba E.G. (1985), Naturalist Inquiry, Sage Publications, Newbury Park, CA.

[22] Llorente I. and Montero R., OpenNebula Project. http://www.opennebula.org/ [Retrieved on 02 July 2010]

[23] March S. T. and Smith G. (December 1995), "Design and Natural Science Research on Information Technology", Decision Support Systems (15:4), pp. 251-266.

[24] Neudesic (2010), Azure ROI Calculator, http://azureroi.cloudapp.net/, [Retriewed on 21 May 2010]

[25] Nunamaker J., Chen M., and Purdin, T.D.M. (Winter 1991), "Systems Development in Information Systems Research", Journal of Management Information Systems (7:3), pp. 89-106

[26] Peiris C., Balachandra, B., and Sharma D. (2010a), 'Cloud Computing Value Proposition: An Interrogation', Proceedings of the Annual International Conference on Cloud Computing and Virtualisation (CCV 2010), pp. 193 – 200, Singapore, May 17-18, 2010.

[27] Peiris C., Balachandran B., and Sharma D. (2010b), "Governance Framework for Cloud Computing", GSTF International Journal on Computing, ISSN 2010-2283, GSTF

[28] Peiris C, Balachandran B & Sharma D (2010c), "Cloud Computing Tipping Point Model", GSTF International Journal on Computing, ISSN 2010-2283, GSTF

[29] Peiris C., Balachandran B., and Sharma D (2010d), "Service Centric Model for Cloud Computing Tipping Point of an ICT Organisation", Proceedings of the Annual International Conference on Cloud Computing and Virtualisation (CCV 2010), pp. 186 – 193, Singapore, May 17-18, 2010.

[30] Peiris C, Balachandran B., and Sharma, D. (2010e) , "Validating and designing a service centric view for C2TP: Cloud Computing Tipping Point model"; Advances in Intelligent Decision Technologies, Vol 4, pp. 423--433, ISBN 978-3-642-14615-2, Springer Heidelberg.

[31] Peiris C., Balachandran B., and Sharma D. (2010f), "C2TP: Service Model for Cloud", International Journal of Cloud Computing, ISSN 2043-9989, Inderscience (in press, accepted on 12th August, 2010)

[32] Simon H.A. (1996), The Sciences of the Artificial (3rd ed.), MIT Press, Cambridge, MA.

[33] Son S. and Sim K.M. (2010), "A Multi -issues Negotiation Mechanism for Cloud Service Reservation", Annual International Conference on Cloud Computing and Virtualization, pp. 123 – 130, Singapore

[34] Spiggle S. (1994), "Analysis and Interpretation of Qualitative Data in Consumer Research", Consumer Reserach, Vol 21, No. 3, p. 491.

[35] Sun Y.L., Perrottb R., Harmerc T., Cunninghamd C., and Wrighte, P. (2010), "An SLA Focused Financial Services Infrastructure", Proceedings of 1st Annual International Conference on Cloud Computing and Virtualization, pp. 59-65, Singapore

[36] ValIT (2010) , ISACA.org , http://www.isaca.org/Template.cfm?Section=Val_IT3&Template=/TaggedPage/TaggedPageDisplay.cfm&TPLID=80&ContentID=51867, [Retrieved on 13 June 2010]

[37] Vouk M. (2008), "Cloud Computing – Issues, research and Implementations", Journal of Computing and Information Technology - CIT 16, 2008, 4, pp. 235–246, doi:10.2498/cit.1001391

[38] Vykoukal J., Wolf M., and Beck R. (2009), "Does Green IT Matter? Analysis of the Relationship between Green IT and Grid Technology from a Resource-Based View Perspective", in Pacific Asia Conference on Information Systems (PACIS).

[39] Youseff L., Seymour K.,You H., Dongarra J., and Wolski R. (2008) The impact of paravirtualized memory hierarchy on linear algebra computational kernels and software, pp.141–152.ACM, 2008

 ISBN: 978-1-61208-106-9

Sociology View on Cloud Computing Value: Actor Network Theory Perspective

Cheng-Chieh Huang/National Taiwan University
Information Management Department of Management School
Taipei, Taiwan
e-mail: d94725007@ntu.edu.tw

Ching-Cha Hsieh/National Taiwan University
Information Management Department of Management School
Taipei, Taiwan
e-mail: cchsieh@im.ntu.edu.tw

***Abstract*—This article argues cloud computing value is generated through dynamic interactions of IT artifacts, services, organizations and their interests. From actor network theory, it illustrates Amazon, Google, and IBM, Microsoft cloud computing value networking and translation, inscription in their actor-network. Further implications such as cloud computing transactions types, standards, economics, and hybrid cloud trends are discussed.**

Keywords-cloud computing; actor network theory; IT value

I. INTRODUCTION

Since Carr published "IT Doesn't Matter" article in Harvard Business Review, causing scholars' attention to reconsider information technology and its business value. Carr described that IT becomes a kind of commodity like water or electricity, IT is no longer valuable. Cloud computing transfers IT artifact into the service like water or electricity, realizes the Carr's concepts. However, cloud computing is also considered as the strategic weapon helping enterprises to lower the cost, increase their competitiveness. Does cloud computing matter or not?

Past literature on IT and business value divided into the two kinds of causality inferences. One is called technology determination, considers that, IT as strategic resource, or innovative tool, the specific IT can create value of organizations [5][6][7]. Another, called organization determination, considers that IT aligned with organization strategy, can increase competitiveness [8][9][10].

But, as regards cloud computing, it seems brings the opportunities of technical innovation, but not all organization can all enjoy the interests immediately. It seems that IT and organization value generation are not for instance, the simple causality influence in this place [2].

In addition, cloud computing is not only the technology innovation but also service innovation. If not considering the service model but prosperities of IT artifact, neglected the important part of cloud computing.

In this article, we argue that cloud computing value is generated through dynamic interactions of IT artifacts, services, organizations and their interests. Using sociology theory, actor network theory (ANT) [14] as a lens, we illustrate Amazon, Google, IBM, and Microsoft cloud computing development cases to demonstrate different business values generation processes. Finally, implications to future cloud computing technology and services development are proposed.

In the following section, we first review the literature of cloud computing and actor network theory. Second, we describe our methodology. Third, the four case stories are illustrated. Fourth, we present analysis and discussion and fifth, we identify contributions, limitations and suggestions for future research.

II. LITERATURE REVIEW

A. Cloud Computing

Cloud Computing refers to the applications or IT resources delivered as services over the internet; also, to the hardware and systems software in that datacenters that provides those services. Although it is popular, but definition of cloud computing is diversity [3][4]. Vaquero et al. give it more careful definition [4]:

> *Clouds are a large pool of easily usable and accessible virtualized resources (such as hardware, development platforms and/or services). These resources can be dynamically re-configured to adjust to a variable load (scale), allowing also for an optimum resource utilization. This pool of resources is typically exploited by a pay-per-use model in which guarantees are offered by the Infrastructure Provider by means of customized SLAs (p.51)*

From this definition, it describes that cloud computing not only the enabled-technology but also the service model.

Retrospect to the evolution history of cloud computing, the cloud computing was the co-evolution from the service innovation and technology innovation (see Fig. 1). That is, while we consider cloud computing, it is not pure the technology artifact but service included.

That is, measuring cloud computing value cannot only derive from their IT characteristics but also its services or economic model. Cloud computing is a kind of techno-economic network (TEN) that Callon mentioned [13].

 ISBN: 978-1-61208-106-9

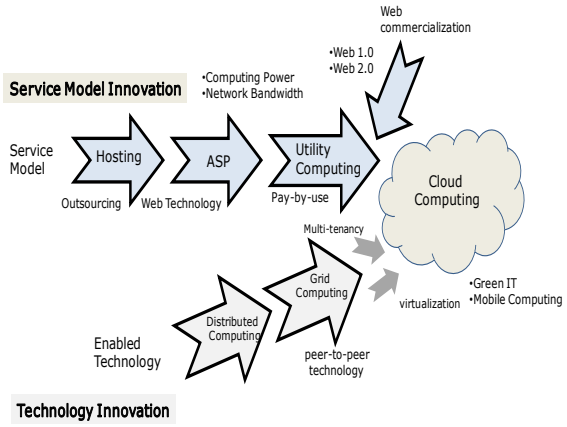


Figure 1. Evolution of cloud computing

Moreover, cloud services and cloud computing technology form a new ecosystem which allied with cloud services providers, cloud infrastructure operators, and cloud technology-enabled providers. Cloud computing includes technology, services, actors and their interests.

B. Actor Network Theory

Actor Network Theory (ANT) was developed in the sociology of science and technology [14]. ANT helps to describe how actors form alliances and involve other actors and use non-human actors (artifacts) to strengthen such alliances and to secure their interests. ANT consists of two concepts: translation and inscription.

When an actor-network is created, consists of four processes of translation [15]:

- Problematization: The focal actors define interests that others may share, establishes itself as indispensable resources in the solution of the problems they have defined. They define the problems and solutions and also establish roles and identities for other actors in the network. As a consequence, focal actors establish an "obligatory passage point" for problem solution which all the actors in an actor-network must pass.
- Interessement: The focal actors convince other actors that the interests defined by the focal actors are in fact well in line with their own interests. Through interessement the developing network creates sufficient incitement to both lock actors into networks.
- Enrollment: Enrollment involves a definition of roles of each of the actors in the newly created actor-network. It also involves a set of strategies through which focal actors seek to convince other actors to embrace the underlying ideas of the growing actor-network and to be an active part of the whole project.
- Mobilization: The focal actors use a set of methods to ensure that the other actors act according to their agreement and would not betray. With allies mobilized, an actor network achieves stability.

In addition to the four stages of translation, the process of inscription is critical to building networks, as most artifacts within a social system embody inscriptions of some interests. As ideas are inscribed in technology and as these technologies diffuse in contexts where they are assigned relevance, they help achieve socio-technical stability.

Through ANT, we can understand how the focal case companies generate values through networks and inscribe their interests into their cloud computing technology and services.

III. METHODOLOGY

In this paper, we use the case study methodology [11] to examine the cloud computing and its business value. We choose four firms, two are internet service firms (Amazon, Google), and two are technology vendors firms (IBM, Microsoft). These four firms are famous with their using cloud computing. We collected documentary data included their cloud computing development histories, news, company reports, successful cases and independent analysis reports such as IDC, Gartner, and Ovum [12]. We also interviewed with their high level managers to talk about their strategies and values of cloud computing. All interview manuscripts are recorded.

Further, we use the events analysis and ANT theory to understand cloud computing value generation processes.

IV. CASE STUDY

A. Amazon

Amazon was established in 1994, the headquarter is located in Seattle of U.S. Amazon relied mainly on engaging in the online bookstore's selling in early days, then flowers, software, electronic goods, toy and other and retailed goods later. Amazon becomes first 500 big online retail businesses in America. The profits generated by Amazon are about 24 billion dollars in 2009.

The development of cloud services of Amazon begins with 2003 and offers web services for its e-commerce partners first. For example, partners want sell music CD in Amazon's online restores, listing the latest music purchase ranks and buyers' comments in order to promote the marketing of its CD and sell. The partners use Amazon's web services offered to develop the promotion web site.

Amazon gradually transfers their internal IT infrastructure to cloud service serve their partners. For example, storage (S3), server computing resources (EC2) and even business process of e-commerce, such as fulfillment process (FWS), payment process (FPS), the personnel matching process (Mechanical Turk).

Through the cloud computing development histories of Amazon, we understand that Amazon early is to offer website designing or developing tools to help its partners to sell on Amazon online store. After developing various kinds of services, Amazon strengthens the whole competitiveness of supply chain operation efficiency with her partners.

 ISBN: 978-1-61208-106-9

B. Google

Google was established in 1998, the headquarters located in California of U.S. Relying on searching engine to attract commercial advertisement, Google becomes the biggest search engine company in the world. The Google earns about 23.6 billion dollars in 2009.

The active one in recent years of Google carries on every tactics overall arrangement, including: merge YouTube, developing cell-phone operating system, Android ,Google Chrome browser, Google Earth and cloud services etc., attracted attention by the market.

Google's clouds services raise from Google API development in 2005. The main purpose of Google API is to let consumers log in their websites frequently, in order to increase the web traffics, strive for the advertiser to put their advertisement on Google websites.

Later, Google begins to develop various types of cloud services, for instance: Google Docs, Google Finance, Google Spreadsheets, Google APE, etc., in order to offer consumers and website designers.

For Google, the search engine traffics equal to money (traffic=$). That is, a series of services are developed and companies merge are to rely mainly on improving the traffics. For example, Google API, Gmail, Google Apps, Google APE, with attracting website designers or consumers in order to attract advertisers to carry on more advertisement . YouTube or Open Social API acquired and merged or linked social community websites in order to get popularity then increasing the traffics. Android operating system cell-phone, Google Chrome can hope for operating system and browser interface on new developing handheld devices of leading factor, ethnicity offering handheld devices to surf the net that has already made the service of Google can be more convenient, connect the flow fetched in order to increase its other equipment.

For Google, cloud services support Google's "traffic equals to money" strategy, that attract more net friends on her search engine, then get more profits from advertisers.

C. IBM

IBM was established in 1924, regard making enterprise information hardware, such as electronic calculator, large-scale mainframe, the first generation of personal computer, etc. Recently, she moves towards more service and software providers to big enterprises.

The development of cloud service was introduced by IBM to help her small independent software vendors (ISVs) partners located worldwide using IBM's server or storage through internet. These ISVs do not need invest hardware/software and can develop software through IBM's platform. Later, IBM developed their cloud computing technology into products that support their enterprise customers to build up the cloud services. Further, IBM provides online cloud services to realize their cloud computing technology.

IBM attempts to use the cloud computing technology products and leverage their consulting services and software implementation experiences in the large-scale enterprise's market and then explores small and medium enterprises and on-line service companies market.

Take her cloud services implementation experiences in UPS for example, IBM combined cloud services with their software implemented in UPS. IBM supported their customers, UPS and also touched UPS's online partners. It is so-called two-sided market strategy includes the large-scale enterprise software service market that IBM has already deeply engaged and new developing medium and small-scale online service companies.

For IBM, cloud services and technology play a bridge role to explore on-line and small medium enterprise (SME) markets opportunities.

D. Microsoft

Microsoft was established in 1983, it is early in leading position which occupies operating system and suit software on the PC of MS-DOS operating system, MS-Office with successful series promptly.

It was about 58 billion dollars that Microsoft earns in 2009. Among them in the suit software / the commercial suit software, the camp of Office series accepts and accounts for all software camp to accept more than ninety percent.

Microsoft realizes that trends moving towards online services, PC or on-promise software is no longer the only choice. So, on one hand, Microsoft defends tenaciously and already has status of operating system, software on PC; on the other hand it can combine the various terminal devices and offer the software, on-line services.

This is the concept of "software plus services" or "3 screens and a cloud" that Microsoft announced her strategy in 2009. For Microsoft, the cloud services or cloud computing technology help her transit to the new "network operating system" smoothly from combing their traditional on-promise software.

V. ANALYSIS AND DISCUSSION

A. Cloud Computing Value Networking

Callon described techno-economic network (TEN) as "a coordinated set of heterogonous actors which interact more or less successfully to develop, produce, distribute and diffuse methods for generation goods and services." From Callon's point of view, the economic value is generated from actors, intermediaries (no-human), translation and their relationships [13].

From the four cases described above, we can understand that different companies interpret that different opportunities of cloud computing, and align their different strategies and try to generate their values.

For them, the cloud computing is not only the technology artifacts, but also services, service models and their customers and partners. It generates value through heterogonous actors with IT, services through networking activities. We call that they "networking IT/service value" (see Fig. 2). Followings are the explanations:

- Amazon: Cloud computing services are innovated from their internal IT, originally support their

business process. Then, Amazon enrolls their e-commerce partner's adopting their web services, and then cloud services embedded in their daily business process. Amazon strengthens their business values through the partners' networks formed by IT and cloud services.

- Google: Google's cloud services, IT products, tools all support their business strategy, the traffic equals to money. Moreover, these services, products leverage each other, and then intensify whole network values.
- IBM: Cloud services originally support IBM's small independent software vendor (ISV) partners to leverage IBM's software/hardware resources through internet. Then, IBM strengthens his technology products, then transfers to cloud services to support their secondary market, online/SME customers. The cloud services bridge a new market.
- Microsoft: Microsoft leverages cloud services to complement her on-promise software, and strength their device product's value. Microsoft tries to enroll her customers to their value network.

B. Translation and Inscription of Cloud Computing

Although these four case companies try to form their actor-network mentioned above, but they still on their different translation stages (see Table I). For example, Amazon enrolls her online store partners into the actor-network and stabilizes network by embedding supply chain processes into their technology. Amazon's online store partners are not easy to betray because of locked operation processes in cloud computing technology and services. Although Google enrolls their customers but still considers how to stabilize their customers using their services.

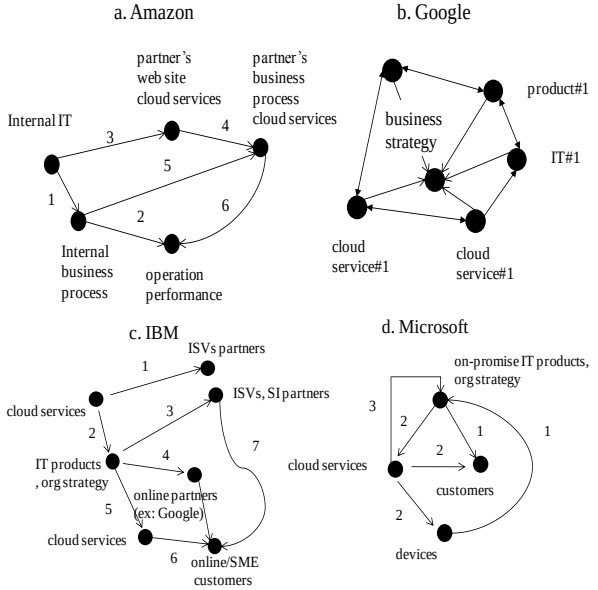


Figure 2. Value networking of case companies

IBM coordinates with Amazon and Google to connect the online SME markets and interest her ISVs partners to join the actor-network. Microsoft is also on his way persuading her ISVs to join actor-network.

Through ANT lens, we also can understand these four foal companies inscribe their ideas and strategies into their cloud computing technology and services (see Table I). For example, Amazon inscribes efficient supply chain processes into their cloud commuting technology for their online store partners. In the future, Amazon will release more technology and services combined with supply chain processes. IBM inscribes their connection ideas into cloud computing technology and services. She tries to use cloud computing technology to connect online SME markets and traditional enterprise markets. Google inscribes everything online into their services and Microsoft tries to defense their on-promise software markets.

C. Implications to Cloud Computing Values and Technology/Services Development

Regarding our cases in this study, these companies' inscribes their ideas or strategies into cloud computing technology or services. Are the properties of technology or service model possible to provide to other actor-network? For examples, Amazon's cloud computing is for online commerce, short, stateless transactions, it will not be suitable for long, stateful traditional enterprise transactions. Moreover, these cloud computing services are easy to scale out to many servers. But the traditional enterprise business services are suitable to single virtualization machine. That is, the cloud computing technology will be more diversity and suitable for their services usage and contexts.

Second, it is not easy to have the de facto or open cloud computing standard, because these focal companies try to develop their own technology or services and generate their actor-network values. But they need expand their networks to enroll more members. That is, brokering services or middleware technology development prop sects well.

Third, it seems the cost down of IT resources the most popular considerations in cloud economics. But from our analysis above, business values are the key for actors to join the network. For example, Amazon's e-commerce partners get value from their operation process efficiency. UPS joins IBM's network, because of integrating her physical logistic process and online services.

Fourth, business values or business models are keys to enroll members to join cloud computing actor-network. That is, the focal companies will blur boundaries between on-line services and cloud computing services (pay-by-usage) model, on-promise software and services. It also means that will hybrid traditional enterprise technology and cloud computing technology as commerce solutions.

 ISBN: 978-1-61208-106-9

TABLE I. TRANSLATION AND INSCRIPTION OF CLOUD COMPUTING

Focal Company	Translation	Inscription
Amazon	Problemaitziaion, Interessement, Envrollment Mobilization	Efficient supply chain process
Google	Problemaitziaion, Interessement, Envrollment	Services on line
IBM	Problemaitziaion, Interessement	Smart Services connection
Microsoft	Problemaitziaion, Interessement	Software plus Services

VI. CONCLUSION

In this article, we discuss the cloud computing value and future technology/service development through ANT theory. We argue that cloud computing values generated through networking of IT, services, organizations and their interests. We also implicate cloud computing technology and services will be more diversity, hybrid and suitable for their services usage and contexts.

This paper is limited to the four company cases analysis. Future research can conduct more companies' cases, and analyze more detail activities and other actors' responses and their inscriptions, translations in their actor-networks.

REFERENCES

[1] N. G. Carr, "IT doesn't matter," Harvard Business Review, pp. 41-49, 2003.

[2] M. L. Markus, and D. Robey, "Information technology and organization change: casual structure in theory and research," Management Science, pp. 583-598, 1988.

[3] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "Above the clouds: A Berkeley view of cloud computing," http://radlab.cs.berkeley/edu/, pp. 1-23, 2009.

[4] L. M. Vaguero, L. Rodero-Merino, J. Caceres, and M. Lindner, "A break in the blouds: Toward a cloud definition," ACM SIGCOMM Computer Communication Review, pp. 50-55, 2009.

[5] K. Lyytinen, and G. M. Rose, "The disruptive nature of information technology innovations: The case of internet computing in systems development organizations," MIS Quarterly, pp. 557-595, 2003.

[6] N. Melville, K. Kraemer, and V. Gurbaxani, "Review: Information technology and organization performance: An integrative model of IT business value," MIS Quarterly, pp. 283-322, 2004.

[7] T. Ravichandran, and C. Lertwongsatien, "Effect of information systems resources and capabilities on firm performance: A resource-based perspective," JMIS, pp. 237-276, 2005.

[8] M. Benaroch, S. Shah, and M. Jeffery, "On the valuation of multistage information technology investments embedding nested real options," JMIS, pp. 239-261, 2006.

[9] J. F. Fairbank, G. Labianica, H. K. Steensma, and R. Metters, "Information processing design, choices, strategy, and risk management performance," JMIS, pp. 293-319, 2006.

[10] T. A. Byrd, B. R. Lewis, and R. W. Bryan, "The leveraging influence of strategic alignment on IT investment: An empirical examination," Information & Management, pp. 308-321, 2006.

[11] R. K. Yin, Case Study Research, Design and Methods, 2nd edition, CA : Sage Publications, 1994.

[12] G. McCulloch, Documentary Research in Education, History and the Social Sciences, London: Routledge Falmer , 2004.

[13] M. Callon, "Techno-economic networks and irreversibility," in A Sociology of Monsters: Essays on Power, Technology, and Domination, J. Law (ed.). New York: Routledge, 1991, pp. 132-164.

[14] M. Callon, and B. Latour, "Unscrewing the big Leviathan", in Advances in Social Theory and Methodology, K. Knorr-Cetina, and A.V. Cicourel, (eds.). London: Routledge & Kegan, 1981, pp. 277–303.

[15] M. Callon, "Some elements of a sociology of translation: Domestication of the scallops and the fishermen of St. Brieuc Bay", iin Power, Action and Belief, J. Law, (ed.). London: Routledge and Kegan Paul, 1986, pp.197–233.

 ISBN: 978-1-61208-106-9

Cloud Credential Vault

Huan Liu
Accenture Technology Labs
50 W. San Fernando St., Suite 1200
San Jose, California, USA
huan.liu@accenture.com

***Abstract*—While cloud computing presents strong value propositions, it also presents significant headaches to enterprise IT departments, including incompatible billing and purchasing process, no policy enforcement and control, and difficult data sharing across users. We describe Cloud Credential Vault – a central repository of cloud access credentials, which is designed to solve these problems facing enterprise IT departments. We describe the Cloud Credential Vault's architecture, design, and how it solves each of the described problems. We also describe its current implementation, where we have already integrated with Accenture's billing system. Our early experience with the Cloud Credential Vault indicats that it can meet the challenges facing the enterprise IT department when managing access to cloud resources.**

***Keywords*-Cloud management, Credential Vault**

I. INTRODUCTION

Cloud computing is already widely used at small and medium businesses. Even large enterprise customers are increasingly evaluating and piloting cloud usage. There are several features of cloud that make it attractive to IT consumers. First, it is on-demand. A user requests a virtual server and the server would be available in a few short minutes. Second, it is pay-per-use. A user no longer needs to buy capital equipment upfront. Third, it is programmable. When an application needs additional capacity, it is a simple API call away. There is no longer the need to over-provision just in case it is needed.

Cloud computing may include many different types of cloud services. One sample service is Infrastructure as a Service (IaaS), such as Amazon EC2, where a user can request Virtual Machines (VM). Other services may include key-value storage services, such as Amazon S3, semi-structured storage services, such as Amazon SimpleDB, or messaging services, such as Amazon SQS.

Although the value propositions of cloud computing is strong, it brings significant disruptions to the current enterprise IT landscape for several reasons. First, its purchasing model does not conform to the standard enterprise purchasing order process. A user can simply pull out a credit card and sign up for cloud services without any IT approval. The charges do not appear in the IT budgeting process until at the end of the month during reimbursement when it is too late. An IT manager has no visibility into the current charges and the spend trend.

Second, an IT department has no control over cloud resources usage and cannot enforce corporate policy. Since a cloud account is under a user's total control, the user could easily abuse the system. For example, a policy may mandate that all data stored in a cloud should be encrypted, but a user can easily ignore the policy, knowing that the IT department has no ability to audit.

Third, a cloud makes it difficult to manage credentials securely. Many cloud services are invoked through a web services API. A user must present valid credentials in order to successfully invoke these APIs. Although this is no different than web services in Service Oriented Architecture (SOA), a cloud makes it more difficult. In a cloud environment, a cloud VM image could be easier shared between users. If the VM needs to access other cloud services (e.g., SQS, SimpleDB, S3), the VM may have to embed the cloud credential. Unfortunately, when the VM image is shared with other users, the credential is inadvertently shared as well. Even if the VM image is never shared, since the image is stored in the cloud, there is the danger that a hacker may hack the image file to obtain the credential. In addition, when the credential is changed (rotating credential regularly is one of the best cloud practices), the VM image must be changed, which is a significant hassle.

Fourth, data sharing in a cloud environment is difficult. When a user needs to share data (either cloud data or VM image) with other users, she must obtain the other users' cloud account ID and then share the data with the ID. Since the mapping from a user to her cloud account ID is maintained manually, it is cumbersome and time consuming to manage data sharing.

In this paper, we describe Cloud Credential Vault (CCV or Vault), which hosts all cloud credentials centrally. By centrally hosting credentials, we enable an IT department to automatically monitor cloud usages and enforce corporate policy. Although CCV is designed to support multiple cloud vendors, our first release currently only supports Amazon services. To be concrete, in the following, we use Amazon services to describe the architecture, design and implementation as needed.

In Section II, we first describe CCV's architecture. Then, we get into more details of CCV's capabilities in Section III. We cover related work in Section IV, and conclude in

 ISBN: 978-1-61208-106-9

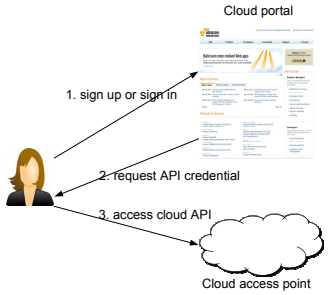


Figure 1. Cloud access model.

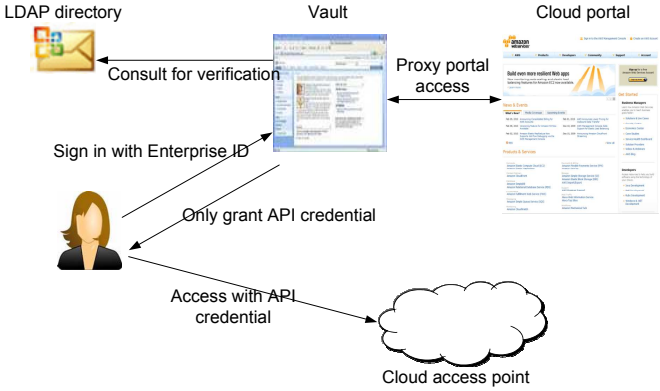


Figure 2. CCV architecture.

Section V.

II. Vault Architecture

In this section, we describe CCV's architecture.

A. Cloud access model

CCV exploits a unique feature of the cloud access model. Since it underpins CCV's design, we first describe the cloud access model, which is shown in Figure 1.

To access cloud resources, a user must first sign up for an account at the cloud vendor's portal page [1]. As part of signing up, the user establishes a username and password pair, which is used to login to the cloud portal. We refer to the username/password pair as the *master credential*, since knowing this pair would allow a person a complete control over the cloud account.

Using the master credential, a user can login to the cloud portal to obtain an *API Credential* – a credential needed to make programmatic API calls to access cloud resources. In Amazon, the API credential consists of an access key and a secret key. In GoGrid, it consists of an API key and a shared secret. In Rackspace, the API credential is an authorization token. Although the authorization token is not obtainable directly from the cloud portal, a user must first login to obtain a username and an API access key, then use them to further obtain the authorization token through an API.

Knowing the API credential is not enough to completely control the cloud account. For example, it is not possible to edit profiles and view/change the billing credit card. However, knowing the API credential is enough to access cloud resources. Although the cloud portal (only accessible through the master credential) typically consists of a GUI dashboard for accessing cloud resources, this functionality can be easily replicated by using the cloud APIs, which only requires the API credential. There are already a large number of third-party GUI console tools available which makes accessing cloud resources easier. For example, ElasticFox [2] is a popular dashboard for Amazon EC2, and S3Fox [3] is a popular dashboard for Amazon S3, both only need the API credential (access and secret key) to function properly.

Many cloud vendors, especially those with programmable APIs, follow this access model: a master credential is used to control the overall account and an API credential is used to access cloud resources. We exploit this separation of credentials in our CCV design.

B. CCV architecture

The basic idea behind CCV is that we split the master credential and the API credential. CCV, which is under the direct control of IT, holds the master credential so that IT can maintain a complete control of the cloud account. When a user is approved to have access to cloud resources, she is handed a unique cloud account which is not shared with others. A user is only given the API credential so that she can access cloud resources, but she is never given the master credential which would have given her total control over the account.

Figure 2 shows CCV's architecture. CCV is a web application, for which users can access directly from their web browser. It integrates with an enterprise's LDAP directory, so that it allows single sign on. Although it currently only works with one administrative domain, we plan to support some form of federated identity, such as that described in [4]. With federation, a CCV can support multiple organizations, making it possible to offer credential management as a service.

CCV interacts with the cloud portal directly. Since a user no longer has access to the master credential, she no longer can perform some functions that she is able to do through the cloud portal. Besides not being able to download the API credential, a user may not be able to perform other functions, e.g., downloading billing statements in the case of Amazon. CCV replicates those functions so that the user still have access to the same information. For Amazon, we screen-scrap the portal page in order to download all billing statements. From the cloud portal, a user could also perform functions that change account settings, such as changing the billing credit card number. Since we do not want the users to view or make those changes, we do not replicate those functions in CCV.

 ISBN: 978-1-61208-106-9

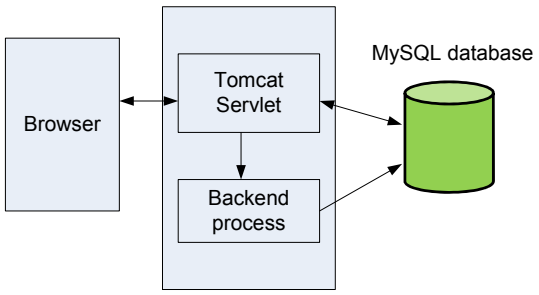


Figure 3. CCV implementation architecture.

Even though not shown, CCV is designed to support multiple cloud providers. In the back end, CCV not only holds the credentials for multiple cloud providers, but it also interacts with each of the cloud portals. For users, CCV can supply cloud accounts from multiple cloud vendors, depending on what a user requests.

A user of CCV can login to CCV to perform functions that she normally would use the cloud portal to perform, such as downloading the API credential, or viewing her cloud usage. Once the user has the API credential, she can access cloud resources directly through the cloud access point using third-party tools. Since only the infrequent action (e.g., viewing statement once a month), but not the more frequent action of accessing cloud resources, goes through CCV, CCV is unlikely to be a performance bottleneck.

Figure 3 shows the current implementation architecture. We use Google Web Toolkit (GWT) to develop the front end browser UI. The backend is implemented as Java Servlet running in a Tomcat container. The backend servlet is responsible for communicating with the frontend through a RPC mechanism. When the frontend invokes RPC calls to retrieve information, the servlet checks (and authenticate if necessary) the user's identity, and then returns the appropriate information authorized for the user.

There is a separate backend process running on the same server. It performs bulk actions that are not part of the web UI's request/response exchange. For example, once a month, the backend process logs into the cloud portal, screen-scraps and downloads the billing statement for each account. In our current implementation, downloading the complete billing statement at Amazon for one account takes roughly 30 seconds, thus it is not suitable to download on-demand when a user requests it. When a user requests for a billing statement that has not been downloaded yet, the servlet passes a request to the backend process, and it then returns immediately. The user is notified that the statement is being updated and that she should wait for a short while before viewing it again.

Screen-scraping simulates a user accessing for information, and a program automatically parses the output for needed information [5]. There are various ways to screen-scrap. For example, we could use the accessibility layer of an operating system to access UI components[6]. However, since all cloud vendors provide a web portal, we choose to use HtmlUnit[7] to parse the returned web page for relevant information.

Besides downloading billing statements, the backend process also performs auditing and policy enforcement. For example, if a policy states that no cloud data should be unencrypted, the backend process periodically takes a sample, and checks if any file conforms to the corporate policy. Since CCV not only has the master credential, but also the API credential, it can easily invoke cloud API to perform the auditing. Section III-B describes the kind of policies that we currently support.

Information downloaded from the cloud portal, such as the billing statement and the credentials, are stored in a MySQL database. For security purpose, all credentials are encrypted before stored in the database. The servlet does not keep any billing or credential data in memory. It always queries the database for the latest information. Thus, the database is our central state storage, which simplifies the synchronization between the servlet and the backend process.

III. Solution details

This section describes the CCV solution in details. In particular, we describe how we address each of the problems described in Section I.

A. Billing integration

One of the goals of CCV is to integrate with an enterprise's internal billing process. CCV accomplishes this goal by acting as a broker between the internal billing system and the cloud.

In the cloud portal, CCV configures each cloud account to use a corporate credit card, which is charged each month by the cloud vendor and then paid directly by the IT department. In Amazon, we enable *consolidated billing* for all cloud accounts under CCV's management. This allows us to benefit from volume discount.

When a user signs up for a cloud account in CCV, she must configures a charge code associated with the cloud account. The charge code is charged each month for the actual cloud usage. In Accenture, the charge code is referred to as the WBS element. Although each company has a different internal billing mechanism, we have designed CCV to be flexible enough that it can easily integrate with a different mechanism.

When a charge code owner logs into CCV, she can see all cloud accounts that are charging to her charge code. She can view billing statements for all those cloud accounts. Since CCV can download partial billing statement when a user or the charge code owner requests, even if it is not the end of the month yet, the charge code owner can view up-to-date spends.

 ISBN: 978-1-61208-106-9

B. Control and policy enforcement

The charge code owner has a full control over the cloud account. She can optionally disable an account (e.g., if the user abuses the account or if the user has left the company), in which case, the API credential is changed so that the user can no longer use it. In addition, the charge code owner can optionally stop all cloud resources usage (stop all servers and/or remove all storage) to stop incurring further charges.

We also plan to support a flexible set of policies that a charge code owner can specify and enforce. However, initially we only support two sample policies.

The first policy states that "no more than x servers are running each day at time y". Both x (a number) and y (a time) are specified by the charge code owner. This policy is designed to catch run-away instances – instances that the user forgot to turn off, which happens frequently in the cloud environment. When enforcing this policy, we start a cron job at the specified time y and use the API credential to query how many EC2 servers are running at the time. If it is more than x, we send an email alert to the charge code owner.

The second policy states that "all cloud data should be encrypted". When this policy is enabled, CCV periodically samples a few files stored in the cloud, and checks whether they are encrypted. For demonstration purpose, we currently only check whether the file is a plain text file. However, in the future, we intend to employ more sophisticated algorithms to detect whether a file is encrypted.

While these two policies are designed to demonstrate the capabilities of CCV, we intend to support a wider range of flexible policies. We are in the process of gathering requirements to understand which set of policies are the most useful.

C. Credential on demand

When a VM needs to access other cloud resources, such as S3, SQS and SimpleDB, a common practice today is to embed the needed API credential inside the VM image. The reason is because it is cumbersome to copy over the API credential every time the VM starts. However, this practice is not secure, for two reasons.

First, the server image could be shared with other cloud users. When other cloud users launch the same image, they would have access to the image owner's API credential.

Second, changing API credential frequently is a cloud best practice, since it minimizes the damage of losing an API credential. Unfortunately, when the API credential is changed, the credential embedded in the VM images remains the same, requiring the image to be recreated again.

Instead of embedding the *image creator*'s API credential, we believe a VM should be entitled to the *image user*'s API credential. CCV provides such a facility to VMs to query the API credential used to launch the VMs in the first place. The VM can request this by querying a URL at the IP address of CCV, but with URI of "/apicredential", e.g., a VM can query https://ccv.com/apicredential.

When a VM queries this API, CCV first has to find the API credential used to launch the VM. We currently iterate over each stored API credential and query Amazon API in sequence to see which API credential the VM was launched under. Although this is inefficient, we have not run into performance problems yet during our pilot trial. We are actively looking into alternative approach to determine the launching credential,

When the launching API credential is found, CCV passes back the API credential to the VM, so that is can start accessing other cloud services such as S3, SimpleDB, and SQS. By providing this mechanism of API credential on demand, we prevent any need to hard-code credential information inside a VM image, thus greatly enhance cloud security.

D. Data sharing

Sharing data across cloud account is cumbersome. A user has to find out about other users' cloud account ID (a 12 digits number in Amazon) and enables sharing with the ID instead of a user name.

In CCV, a user can share cloud data and server images with a user name or a group alias. The group aliases are from the LDAP directory. When a user chooses to share with a group (e.g., HR.global group), CCV looks up in the LDAP directory to expand the group into a list of user names. From the list of user names, CCV then expands it into a list of cloud account IDs by checking the cloud account IDs that belong to each user. It then shares the data or image with the list of account IDs. Even though CCV still uses the cloud's native capability to share data with an account ID, the user operates at a much higher abstraction layer, sharing with a user or a group of users.

The group membership in LDAP directory may change over time, e.g., new members joining HR.global or existing members leaving HR.global. CCV periodically checks the group membership and adjusts the permission as needed.

IV. Related work

RightScale[8] and EnStratus[8] all address the same management challenge facing enterprises trying to adopt cloud computing. However, both of them take a different approach than ours. They use one cloud account to support a whole enterprise, and build an interface on top to multiplex multiple cloud users through the same cloud account. Although it has the ability to limit a user to a subset of cloud functionalities, it has two disadvantages. First, their interfaces have to be very scalable since all cloud access go through those interfaces. Second, they cannot perform accurate accounting between different projects. For example, a VM could consume significant bandwidth charges, but since its bandwidth usage does not go through the management interfaces, it is

 ISBN: 978-1-61208-106-9

not possible for those interfaces to meter and bill projects for those bandwidth charges.

MyProxy[9][10] is a repository of X.509 proxy certificates[11]. CredEx [12] extends MyProxy to further support heterogeneous authentication methods. These repositories all treat the credentials as opaque, whereas we take advantages of the API credentials to enable control and auditing.

Amazon cloud manages SSH public key in a similar mechanism as we used for passing the API credential to an image. The SSH public key is available at a fixed IP address (169.254.169.254) and a fixed URI (/latest/meta-data/public-keys). To pass the API credential, we also use a fixed IP address (CCV's IP address) and a fixed URI (/apicredential).

V. Conclusion and future work

We have presented the architecture, design and implementation of the Cloud Credential Vault – a central repository of cloud credentials. By centralizing the credentials and by separating the master credential from the API credential, CCV solves many management problems facing enterprises that are adopting cloud computing.

We have only completed a prototype implementation supporting only one cloud vendor (Amazon). Beyond supporting more cloud vendors, there are also several open questions that we need to address to make CCV more efficient. First, we need a more efficient mechanism to look up the API credential used to launch a particular VM. Second, we need to define a more comprehensive set of policies to support. Third, we are also looking at more efficient ways to monitor group membership changes so that we can adjust data sharing permission as needed.

References

[1] Amazon Web Services, "Amazon Web Services Portal," http://aws.amazon.com, 08.23.2010.

[2] Elasticfox, http://sourceforge.net/projects/elasticfox, 08.23.2010.

[3] Suchi Software, "S3fox," http://www.s3fox.net/, 08.23.2010.

[4] E. R. Mello, M. S. Wangham, J. Silva Fraga, E. T. Camargo, and D. Silva Böger, "A model for authentication credentials translation in service oriented architecture," pp. 68–86, 2009.

[5] B. Myers, "User interface software technology," *ACM Comput. Surv.*, vol. 28, no. 1, pp. 189–191, 1996.

[6] M. Grechanik, K. Conroy, and K. S. Swaminathan, "Creating web services from gui-based applications," in *Proc. SOCA*, 2007.

[7] Htmlunit, http://htmlunit.sourceforge.net, 08.23.2010.

[8] Rightscale, http://www.rightscale.com, 08.23.2010.

[9] J. Novotny, "An online credential repository for the grid: Myproxy," in *Proceedings of the Tenth International Symposium on High Performance Distributed Computing (HPDC-10), IEEE*. Press, 2001, pp. 104–111.

[10] J. Basney, M. Humphrey, and V. Welch, "The myproxy online credential repository," *Software: Practice and Experience*, vol. 35, pp. 801–816, 2005.

[11] S. Tuecke, V. Welch, D. Engert, L. Pearlman, and M. Thompson, "Internet X.509 public key infrastructure (PKI) proxy certificate profile." RFC 3820 (Informational), 2004. [Online]. Available: http://www.ietf.org/rfc/rfc3820.txt

[12] D. D. Vecchio, M. Humphrey, J. Basney, and N. Nagaratnam, "Credex: User-centric credential management for grid and web services," *Web Services, IEEE International Conference on*, vol. 0, pp. 149–156, 2005.

 ISBN: 978-1-61208-106-9

Model-Based Migration of Legacy Software Systems to Scalable and Resource-Efficient Cloud-Based Applications: The CloudMIG Approach

Sören Frey and Wilhelm Hasselbring
Software Engineering Group
University of Kiel
24118 Kiel, Germany
{*sfr, wha*}*@informatik.uni-kiel.de*

***Abstract*—The paper describes the model-based approach CloudMIG. Cloud computing supplies software, platforms, and infrastructures as a service (SaaS, PaaS, and IaaS, respectively) over a network connection. Cloud providers frequently offer the services according to the utility computing paradigm. Therefore, cloud computing provides means for reducing over- and under-provisioning through enabling a highly flexible resource allocation. Running an existing software system on a cloud computing basis usually involves extensive reengineering activities during the migration. Current migration approaches suffer from several shortcomings. For example, they are often limited to specific cloud environments or do not provide automated support for the alignment with a cloud environment. We present our model-based approach CloudMIG which addresses these shortcomings. It aims at supporting SaaS providers to semi-automatically migrate existing enterprise software systems to scalable and resource-efficient PaaS and IaaS-based applications.**

***Keywords*-Approach CloudMIG; Cloud Computing; Model-based software migration to cloud-based applications; Resource-efficient cloud-based applications.**

I. INTRODUCTION

Most enterprise applications' workload underlies substantial variations over time. For example, user behavior tends to be daytime-dependent or media coverage can lead to rapidly increasing popularity of provided services. These variations often result in over- or under-provisioning of data center resources (e.g., #CPUs or storage capacity). Cloud computing provides means for reducing over- and under-provisioning through supplying elastic services. Thereby, the conformance with contractually agreed service level agreements (SLAs) has to be ensured [1]. Considering legacy software systems, is there a way established enterprise applications can benefit from present cloud computing technologies? For reasoning about this issue, it is useful to clarify the main participants in providing and consuming cloud computing services. According to [2], three different roles can be distinguished. SaaS providers (cloud users) offer software services which are being utilized by SaaS users. For this purpose, the SaaS providers may build upon services offered by cloud providers (cloud vendors). In the following, we will employ the terms SaaS user, SaaS provider, and cloud provider.

Newly developed enterprise software may easily be designed for utilizing cloud computing technologies in a greenfield project. Though, SaaS providers may also consider to grant responsibility of operation and maintenance tasks to a cloud provider for an already existing software system. Running established enterprise software on a cloud computing basis usually involves extensive reengineering activities during the migration. Nevertheless, instead of recreating the functionalities of an established software system from scratch for being compatible with a selected cloud provider's environment, a migration enables the SaaS provider to reuse substantial parts of a system. The number of system parts which might be migrated is dependent on the weighting of several parameters in a specific migration project. For example, implications concerning the performance or structural quality metrics regarding the resulting software architecture can be taken into account. Furthermore, aligning a software system to a cloud environment's special properties during the migration process has the potential to increase the software system's efficiency. For example, a reengineer could decide to prefer utilization of certain resources according to their pricing. Considering such kinds of favorable resource utilization and a cloud environment's specific scalability mechanisms can improve overall resource efficiency (e.g., according to the aforementioned prioritization) and scalability. However, there are several major obstacles which can impede such migration projects. Current approaches are often limited to specific cloud environments or do not provide automated support for the alignment with a cloud environment, for instance. In this work, we propose our model-based approach CloudMIG which addresses these shortcomings and focuses on the SaaS provider perspective. The semi-automated approach aims at assisting reengineers in migrating existing enterprise software systems to scalable and resource-efficient PaaS and IaaS-based applications (e.g., see [3]).

The remainder of the paper is structured as follows: Section II describes the shortcomings of existing approaches. Our approach CloudMIG is presented in Section III, before Section IV draws the conclusions and outlines the future work.

 ISBN: 978-1-61208-106-9

II. Current Shortcomings

Migrating typical enterprise software to a cloud-based application usually implies an architectural restructuring step. However, knowledge about the internal structure of the existing software system is often insufficient and therefore an architectural model has to be reconstructed first. The architectural model serves as a starting point for restructuring activities towards a cloud-compatible target architecture, which most often has to be created manually. This often is not an easy task, as construction of the advanced architecture usually presumes profound comprehension of the existing one. Furthermore, the target architecture must comply with the specific cloud environment's offered resources and imposed constraints, for example application frameworks and limitations of programming interfaces, respectively. A mapping model that describes the relationships between system parts of the status quo and the target architecture is needed as well. Future workload in combination with the target architecture arrangement will determine resource utilization of the cloud environment during operation. As most cloud providers follow the paradigm of utility computing and therefore charge resource utilization on a pay-as-you-use basis, the arrangement of the target architecture has a direct impact on the operational costs. Moreover, running an application in a cloud environment does not solve scalability issues per se. For example, an IaaS-based application often needs to have built-in self-adaptive capabilities for leveraging a cloud environment's elasticity.

Shortcomings of today's migration projects from typical enterprise software to cloud-based applications can therefore be summarized as follows:

- ***S1*** **Applicability:** Solutions for migrating enterprise software to cloud-based applications are limited to particular cloud providers.
- ***S2*** **Level of automation:** The target architecture and the mapping model often have to be built entirely manual. Additionally, the target architecture's violations against the cloud environment's constraints are not identified automatically at design time.
- ***S3*** **Resource efficiency:** Various migrated software systems are not designed to be resource-efficient and do not leverage the cloud environments' elasticity, because even transfering an established application to a new cloud environment is a challenging task itself. Furthermore, means for evaluating a target architecture's dynamic resource utilization at design time are most often inadequate.
- ***S4*** **Scalability**: Automated support for evaluating a target architecture's scalability at design time is rare in the cloud computing context.

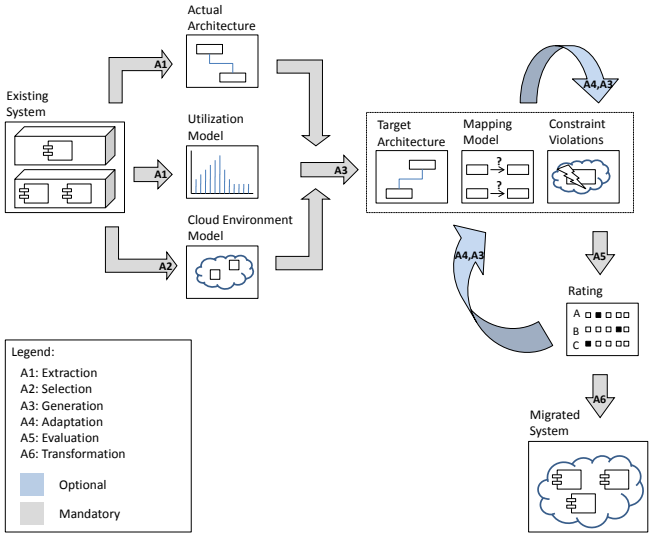


Figure 1. CloudMIG Overview.

III. The Approach CloudMIG

CloudMIG is composed of six activities for migrating an enterprise system to a cloud environment. It provides model-driven generation of considerable parts of the system's target architecture. Feedback loops allow for further alignment with the specific cloud environment's properties and foster resource efficiency and scalability on an architectural level. Fig. 1 outlines the approach. Its activities (A1-A6) are briefly described in the following including the involved models.

A. Activity A1 - Extraction

CloudMIG aims at the migration of established enterprise applications. Usually, the architecture of software systems tends to erode over time. Therefore, initially envisioned architectures frequently diverge from actual implementations. The knowledge about the internal structure is often incomplete, erroneous, or even missing. As CloudMIG utilizes a model transformation during generation of its target architecture (cf. A3), a representation of the software system's actual architecture has to be available first. Concerning this issue, an appropriate model is extracted by means of a software architecture reconstruction methodology. We propose OMG's Knowledge Discovery Meta-Model[1] (KDM) for building a suitable meta-model.

For leveraging the commonly applied utility computing paradigm, the target architecture has to be laid out resource-efficient and elastic. Therefore, CloudMIG includes the extraction of an established software system's utilization model acting as a starting point. The utilization model (resp. its meta-model) includes statistical properties concerning user behavior like service invocation rates over time or average submitted datagram sizes per request. Relevant

[1] http://www.omg.org/spec/KDM/ (Accessed August 16, 2010)

 ISBN: 978-1-61208-106-9

information can be retrieved from various sources. For example, considering log files or instrumenting the given system with our tool Kieker[2] for setting up a monitoring step constitute possible techniques. Furthermore, the utilization model contains application-inherent information related to proportional resource consumption. Metrics of interest could be a method's cyclomatic complexity or memory footprint. We propose OMG's Software Metrics Meta-Model[3] (SMM) as a foundation for building the related meta-model.

B. Activity A2 - Selection

Common properties of different cloud environments are described in a cloud environment meta-model. Selecting a cloud provider specific environment as a target platform for the migration activities therefore implies the selection of a specific instance of this meta-model. For example, this meta-model comprises entities like VM instances or worker threads for IaaS and PaaS-based cloud environments, respectively. As a result, for every cloud environment which shall be targeted with CloudMIG a corresponding meta-model instance has to be created once beforehand. Transformation rules define possible relationships to the architecture meta-model.

We plan to attach further information related to scalability issues to the included entities, which can be configured by the reengineer in activity A4. For example, VM instances could provide hooks for controlling their lifetime dependent on dynamic resource utilization during runtime. Furthermore, the meta-model includes constraints imposed by cloud environments restricting the reengineering activities. For example, the opening of sockets, the manual spawning of threads, or the access to the file system are often constrained.

C. Activity A3 - Generation

The generation activity produces three artefacts, namely a target architecture, a mapping model, and a model characterizing the target architecture's violations of the cloud environment constraints. The latter lists the features of the target architecture which are non-conform with the cloud environment's specification. These constraint violations explicitly highlight the target architecture's parts which have to be redesigned manually by the reengineer (cf. A6). The mapping model assigns elements from the actual architecture to those included in the target architecture. Finally, the target architecture constitutes a primary artefact. It is realized as an instance of the cloud environment meta-model. We propose three phases P1-P3 for the generation of the target architecture.

P1 - Model transformation: The phase P1 produces an initial assignment from features of the existing architecture to cloud-specific features available in the cloud environment model. The initial assignment is created applying a model-to-model transformation according to the transformation rules included in the cloud environment model (cf. activity A2).

P2 - Configuration: The phase P2 serves as a configuration of the algorithm used for obtaining a resource-efficient feature allocation in phase P3. During P2, a reengineer may adjust rules and assertions for heuristic computation (cf. P3). A rule could be formulated like the following examples: "Distribute the five most frequently used services to own virtual machines" or "The server methods responsible for at least 10% of overall consumption of the CPU time shall be moved to client side components if they do not need access to the database". An exemplary assertion could be: "An existing component must not be divided in more than 3 resulting components". It is intended to provide a set of default rules and assertions. In addition to that, the reengineer will be given the possibility to modify them either via altering the regarding numerical values or applying a corresponding DSL. In both cases, the rules and assertions have to be prioritized after their selection. Hereby, the reengineer determines their significance during execution of P3. This means that architectural features which are related to higher-weighted rules will be considered priorly for assignment and therefore have a stronger impact on the further composition of the target architecture. Furthermore, a reengineer may pin architectural features. This prevents the reallocation of previously assigned architectural features to other target architecture components in phase P3.

P3 - Resource-efficient feature allocation: The phase P3 improves the initial assignment of architectural features generated in phase P1 referring to resource-efficiency. Therefore, the formulated rules are utilized and the compliance of the resulting architecture with the defined assertions is considered. There exists an enormous number of possible combinations for assigning architectural features. Efficiency improvements for one resource can lead to degradation for other resources or impair some design quality attributes. For example, splitting a component's parts towards different virtual machines can improve relative CPU utilization, but may lead to increased network traffic for intra-component communication and a decreased cohesion. Additionally, those effects do not necessarily have to move on linearly and moreover, the interrelations are often ambiguous as well. Therefore, we propose application of a heuristic rule-based approach to achieve an overall improvement. A potential algorithm is sketched in Fig. 2 and it works as follows.

The rules are considered successively according to their priority. Thus, rules with higher priorities are weighted higher and have a stronger impact on the generated target architecture. The selection criterion of a rule is defined to deliver a set of scalar architectural features. All possible subsets of the set are rated respective to the quality of

[2]http://kieker.sourceforge.net/ (Accessed August 16, 2010)
[3]http://www.omg.org/spec/SMM/ (Accessed August 16, 2010)

 ISBN: 978-1-61208-106-9

1: $F_{Pinned} \leftarrow$ Pinned architectural features
2: $R \leftarrow$ All rules
3: $A \leftarrow$ All assertions
4: $R_{Sort} \leftarrow$ Sort R descending by priority
5: $F_{AllAffected} \leftarrow F_{Pinned}$
6: **for all** r in R_{Sort} **do**
7: $F_r \leftarrow$ All architectural features delivered by r's selection criterion
8: $P_r^F \leftarrow$ Power set of F_r
9: $Score \leftarrow$ New associative array
10: **for all** p_r^F in P_r^F **do**
11: $Score[p_r^F] \leftarrow$ Rate p_r^F
12: **end for**
13: $Score_{Sort} \leftarrow$ Sort $Score$ descending by score
14: $Score_{Sort}^{Keys} \leftarrow$ Keys of $Score_{Sort}$
15: **for all** p_r^F in $Score_{Sort}^{Keys}$ **do**
16: $F_{FormerlyAffected} \leftarrow p_r^F \cap F_{AllAffected}$
17: $F_{NeedReallocation} \leftarrow$ Elements of $F_{FormerlyAffected}$ that need reallocation conc. r
18: **if** $F_{NeedReallocation} == \emptyset$ **then**
19: $A_{HigherPrio} \leftarrow$ All $a \in A$ with higher priority than r
20: **if** $\nexists a \in A_{HigherPrio}$ with r violates a **then**
21: Apply rule r to all features in p_r^F
22: $F_{AllAffected} = F_{AllAffected} \cup p_r^F$
23: **end if**
24: **end if**
25: **end for**
26: **end for**

Figure 2. Rule-based heuristics for creating a resource-efficient feature allocation.

the target architecture that would result, if the features in the subset would be assigned correspondingly. This aims at considering interdependencies at the level of a single rule. For regarding interdependencies on an inter-rule level, the formulated assertions are taken into account. A rule is only applied if the reengineer did not formulate an assertion with a higher priority that would be violated after the rule's execution. Furthermore, the rule is applied to all mentioned subsets in order of their score. However, the rule is only utilized if no rearrangement of features is necessary whose subset was rated higher. The same applies to assignments that would lead to rearrangement of features that were placed by rules of higher priority or formerly pinned features.

D. Activity A4 - Adaptation

The activity A4 allows the reengineer to adjust the target architecture manually towards case-specific requirements that could not be fulfilled during generation activity A3. For example, the generation process might not have yielded an expected assignment of a critical component. Furthermore, for leveraging the elasticity of a cloud environment, the reengineer might configure a capacity management strategy by means of utilizing the hooks provided by entities contained in the cloud environment meta-model (cf. A2).

E. Activity A5 - Evaluation

For being able to judge about the produced target architecture and the configured capacity management strategy, A5 evaluates the outcomes of the activities A3 and A4. The evaluation involves static and dynamic analyses of the target architecture. For example, the metrics LCOM or WMC can be utilized for static analyses. Considering the target architecture's expected runtime behavior, we propose to apply a simulation on the basis of CloudSim [4]. Thus, we intend to contribute a transformation from CloudMIG's cloud environment meta-model to CloudSim's simulation model.

F. Activity A6 - Transformation

This activity comprises the actual transformation of the enterprise system from the generated and improved target architecture to the aimed cloud environment. No further support for actually accomplishing the implementation is planned at this time.

IV. Conclusion and Future Work

We presented an early work concerning our model-based approach CloudMIG for migrating legacy software systems to scalable and resource-efficient cloud-based applications. It concentrates on the SaaS provider perspective and facilitates the migration of enterprise software systems towards generic IaaS and PaaS-based cloud environments. CloudMIG is intended to generate considerable parts of a resource-efficient target architecture utilizing a rule-based heuristics. The future work focuses on the realization, improvement, and evaluation of CloudMIG's target architecture generation and evaluation activities (A3 and A5, respectively).

References

[1] W. Iqbal, M. Dailey, and D. Carrera, "SLA-Driven Adaptive Resource Management for Web Applications on a Heterogeneous Compute Cloud," in *CloudCom*, ser. Lecture Notes in Computer Science, M. G. Jaatun, G. Zhao, and C. Rong, Eds., vol. 5931. Springer, 2009, pp. 243–253.

[2] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "Above the Clouds: A Berkeley View of Cloud Computing," EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2009-28, Feb 2009.

[3] L. M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner, "A break in the clouds: towards a cloud definition," *SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 1, pp. 50–55, 2009.

[4] R. N. Calheiros, R. Ranjan, C. A. F. D. Rose, and R. Buyya, "CloudSim: A Novel Framework for Modeling and Simulation of Cloud Computing Infrastructures and Services," *CoRR*, vol. abs/0903.2525, 2009.

 ISBN: 978-1-61208-106-9

Understanding the Relationship Between SDP and the Cloud

Stéphane H. Maes
Applications and Software, Huawei, Santa Clara, CA, USA
smaes@huawei.com

***Abstract* — Notions of SDP (Service Delivery Platform) has been widely used in Telcos albeit without a common industry understanding. SDP projects have spearheaded next generations of Telco service provider (SP) services and their forays into new business models. Recently, Telco players tried very actively to understand how to take advantage of Cloud Computing to improve efficiencies or open new opportunities for them. Today most initial activities have often started disconnected from Consumer, VAS and Enterprise activities where SDPs are used. The main contributions of this paper are in explaining how to understand SDPs in the context of Cloud Computing and in positioning Cloud Computing and SDPs from an architecture for Telcos that want to reduce cost and complexity and increase efficiencies, scalability and availability of existing SDP and communication services as well as explore new business models with differentiators that could favor Telco SPs over other more conventional cloud providers. Such recommednatiosn and blueprints have never been discussed before as far as we know.**

Keywords-component: Service Delivery Platforms, SDP, Cloud Computing, IaaS, PaaS, SaaS, Telecommunications, Telcos, SDF, OSS, BSS, Services

I. INTRODUCTION

Telcos (i.e., Telecommunications operators or Telco SPs in this paper) have deployed many variations of SDPs over the last decade, sometimes multiple SDPs within a same domain, to efficiently offer services including core communications, collaboration and multimedia services, content sales and delivery, exposure to third parties and reselling, etc. Initially some operators have deployed SDPs as one platform per service, network (technology) or target market. Today, one can see many SDP consolidations around one platform across all networks and for all services, including examples of multinational SDPs that allow multi-property operators to reduce cost and time to market across their multiple local subsidiaries as well as increase the application addressable market and therefore attractiveness to third party developers.

Still, the absence of similar and sustained SDPs across enough Telcos has hurt the attractiveness of SDPs for developers when compared to the opportunities opened to them by the internet or the recent applications stores for smartphones. The omnipresence of the Internet / web and its fundamentals impacts on the business of Telcos, is today forcing Telcos to better compete or cooperate with Internet Service Providers. In fact, one can expect that that many Telco SPs today will eventually morph their service business into similar "SoftCos" (i.e., not really distinguishable from many of the Internet service providers like Google, Yahoo, Facebook and many others).

In the context, the successes or promises of Cloud Computing initiatives driven by companies like Amazon, Google, Yahoo, Salesforce.com and Microsoft as well as the offerings of companies like IBM, Oracle, VMWare and many open source projects have caught the attention of Telcos. Could it really be an ideal opportunity for Telcos to develop new business models with significant credibility, differentiators and hence chances of success? Certainly many Telcos realize that the market is moving fast but they still lack validation of the opportunities…

Today many Telcos have some Cloud Computing initiatives, proof of concepts, pilots, initial offerings or even production services [7]. The industry seems to have learned from the mistakes of wall garden Telco-only vertical solutions that failed repeatedly in the past (e.g. IMS (IP Multimedia Subsystem) [3,4]): most initiatives are built or extend on what we may call as IT or Internet clouds … as examplified by Amazon AWS [6] (Most projects reuse AWS EC2 or S3 services and possibly a few others).

For Cloud Computing Audience, we believe that it is important to summarize what an SDP really is before discussing the relationship to the cloud and recommending how "IT or internet compatible clouds" can be used or offered by Telcos with differentiated features that exploit Telcos strengths.

II. UNDERSTANDING CLOUD COMPUTING AND TELCOS

In this paper, Cloud Computing refers to Figure **1**.

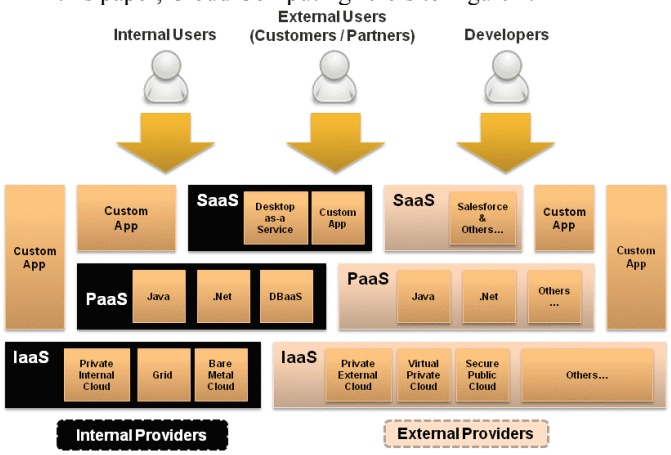


Figure 1 – Anatomy of Cloud Computing. Figure inspired from [1]

Cloud Computing enabless "capabilities available on demand" or "on demand offerings" that can respectively support internal need or what a customer uses and therefore pays for. These include: 1) IaaS (Infrastructure as a Service) that can provide computing infrastructure (e.g. virtualized OS with underlying computing HW (CPU) and storage). 2) PaaS (Platform as a Service) which include a control plane and an application plane with development and execution environments. The associated development tools can also be hosted in the cloud (and for example accessed through a browser). With PaaS, developers can build (web) applications on its SEE/SCE (Service Execution Environment / Service Creation Environment) without installing any tools on their computer and then deploy those applications without any specialized systems administration skills. The control plane can provide a combination of features like application scalability, application plane and infrastructure usage management shared across many applications (GAE (Google App Engine), Force.com) or customizable per application (~ like with OSGI bundles an application can be bundle with the SEE bundles that it requires). For service providers of complex or critical applications (like Telcos), the latter is recommended as it ensures easier management and performance assurance while the former is more for "consumer" or simpler less critical applications. 3)

 ISBN: 978-1-61208-106-9

Software as a Service: software that is deployed over the internet or intranet. With SaaS, a provider rents an application to customers on demand, through a subscription or a "pay-as-you-go" model.

As described in Figure 1, a Cloud Computing solution can be built on multiple "clouds" configurations: private cloud (i.e., in house on demand infrastructure), public cloud (provided by other cloud service providers) or combinations of the above. The latter illustrates especially well how Cloud Computing can allows a service provider to handle peak requirements without having to allocate or invest in infrastructure that would otherwise remain idle, hence reducing CAPEX requirements. With a PaaS, OPEX costs are also further reduced by consolidating the development environment and the life cycle management of the applications developed on it.

Offering IaaS, PaaS and SaaS open new business models to any service providers. PaaS today are only closely closely-held and only available as a Public Cloud service. No full fledge PaaS exist as a middleware product offering yet. None offer yet the customization mentioned above.

III. CLARIFYING THE NOTION OF SDP

In the absence of an industry accepted definition of SDP, we proposed to define a SDP as IT SOA middleware with Telco / communications features, capabilities and performances (see Figure **2**) [5]. A good blueprint of how it should be implemented is provided by the OSE (OMA Service Environment) and related OMA and ITU work and specifications [2].

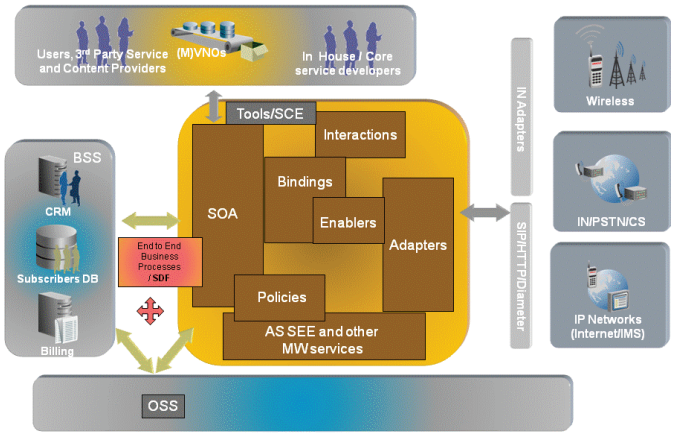


Figure 2 – Blueprint and positioning in a Telco Environment (or generic Service provider environment) of a SDP as Middleware + Telco (in fact communications as this works also on internet for ISPs) functions, capabilities and performances [5].

Accordingly, the SDP IT middleware provides communication functions (e.g. enablers and SEE/SCE like SIP AS (e.g. JSR 289), etc) in a SOA IT middleware (e.g. JEE). The underlying network capabilities are exposed as enablers and controlled by these enablers through adapters that abstract them from the underlying network technologies. Applications, in house or developed by third parties are implemented by composition of enablers and / or within the SEE/SCE. This way, applications can be network technology and equipment vendor independent, future proof and convergence, continuity and FMC services are trivial to implement [4]. Enablers can be used to expose (Telco) backend applications like BSS (Business Support Systems) e.g. CRM, Billing Systems and BI and like OSS (Operating Support Systems) like network and system management, activation, provisioning. In this paper, in the context of a SDP and according tour SDP recommendation and blueprint [2, 4, 5], resources denote the system exposed and controlled via enablers or SEE/SCE (e.g. network elements like location server, SMSC, SIP router, IN call control, HSS, HLR, media servers, IVR/Voice servers etc, or OSS, BSS functions). SDPs can also be integrated with OSS and BSS via end to end SOA business processes as described in [5].

With a SDP, Telcos can:

- Repurpose, increase, improve or rapidly and efficiently develop, with IT software practices, current and next generation core services as well as new communications services across Telco and Internet Networks.
- Develop many content (and application or service) delivery services that allows operators to play new roles in the Internet value chain by becoming reseller, content aggregators, content distributors or even content providers towards their users.
- Asset exposure (to third party)
  - Following the OSE blueprint [2], SDP can be used to expose through interfaces of enablers the operator's assets and capabilities, like features from the operator's network, OSS or BSS to applications, especially to third party applications. The interfaces typically accommodate web services (e.g. SOAP) and web 2.0 bindings (e.g. REST) and the SDP ensures that the exposure is "controlled" by enforcing policies and SLAs on all requests and responses to and from the enablers and containers.
  - This enables operators to create "two sided businesses". Conventionally this was for consumer services, but recently two sided businesses have proven also very successful to increase the role, and revenues, of operators with enterprises. Indeed, it allows service providers to resell third party services to enterprises bundled with their own communications services, therefore increasing revenue opportunities for operators and facilitating the distribution for third parties.
  - Both for consumer or enterprise markets, policy controlled exposure and two sided business models can enable new sources of revenue for operator where they can for example:
    - Monetize their assets
    - Share revenue with third party application / service providers
    - Create businesses for other third party to sell "enablers".
    - The model can be revenue sharing or reselling agreement etc.
  - With enterprises, the approach allows also operators to insert themselves in the enterprise value chain
- Enablers expose relevant function ranging from communications functions like call control, media control, multimedia messaging, contextual functions like location, presence, OSS functions like device management or system provisioning or BSS functions like user or subscriber profile management, subscription management or account management, payment etc. They may also denote reusable functions (SOA) or MW exposed services.
  - Some enablers open the possibility of new businesses for operators like for example:
    - Payment and account management can enable operators to allow third party to bill services against the bill (pre or post-paid) and therefore provides alternatives to services like pay pal.
    - Profile information or derived recommendations or business intelligence can be communicated (possibly filtered to respect privacy, preferences or regulation)
- Policy enforcement. A SDP provides advanced policy decision capabilities. The can be used to provide ease of control of many other assets like:
  - Policy based differentiated traffic QoS
  - Converged charging as policy enforcement
  - Traffic control based on policy enforcement
  - Etc.

 ISBN: 978-1-61208-106-9

- Anything else: as an open platform, operators can use the service creation and execution of the SDP middleware for any other need that they have, thereby enabling them to develop new services on par with the IT and internet players.
  - For example, developing recommendations, "ad insertion" and "ad warehousing" solutions that can be used to:
    - Subsidize services to better compete against the Internet "free" services that are similarly subsidized but differentiated with the addressable channels, applications that can support ad insertions and accurate ad targeting.
    - Maintain or create relationships with advertiser instead of giving them up to Internet advertisement warehouse like Google, Yahoo etc.

With the above, operators are able to offer more efficiently services and better compete or partner with other IT/ Internet service providers. Tellcos can also create new businesses. For example:

- Providers of financial services like mobile banking, payments
- Providers of hosted platforms for third party services
- Aggregators or federation of web 2.0 and social network services.

These new services and some of the other services described above might be considered to be offered to "Internet users" instead of just the current Telco subscribers... This might truly perfect the transition of the Telco SP to SoftCos (i.e., Telco service providers looking better like Internet service providers).

It is worth nothing also that SDP as IT/SOA middleware with Communications capabilities are also suitable platforms for other Service providers or Enterprises aiming at developing or offering communications related services…

IV. SDP, SDF AND OSS/BSS

SDF (Service Delivery Framework) denotes SOA integration of OSS, BSS and SDP and the services running over it. It encompasses patterns like EIA (Enterprise Integration Architecture), AIA (Application Integration Architecture) and TMF SDF [1].

The integration enables the automation of end-to-end business processes across the OSS, BSS and SDP, like "concept to cash", "trouble to resolve", etc. Modern SDPs [5] provide such integration framework and business processes so that it further reduces the time to market of new services and automates all business processes surrounding businesses built around the services.

V. TELCO SERVICE PROVIDERS AND CLOUD COMPUTING

Telcos like many other players in other industries have observed the trends of Cloud Computing and the emergence of new players as providers of infrastructure, platform or software on demand [7]. It is has been argued that the business models around Cloud Computing play well along some of the core expertises of Telco like:

- Offering (often to other businesses) reliable infrastructure (and services) integrated with reliable supporting OSS and BSS.
- Bundling capabilities from their own assets like network QoS, Prepackaged OSS, BSS and end to end business processes.
- Offering communications services that can be considered as precursors of SaaS business models (especially towards enterprise customers) albeit often today without the "on demand" / elastic scalability characterizing Cloud Computing.

Therefore, many Telcos envisage creating successful Cloud Computing based businesses (See [7] for details). In fact many Telcos have already started to explore or provide such services. However, these initiatives have been so far mostly separated from their core services businesses and often driven by other departments than conventional Telco departments like IT, OSS, BSS and Network. With notable exceptions where Cloud Computing is provided as internal IT services provided by theTelco's IT department.

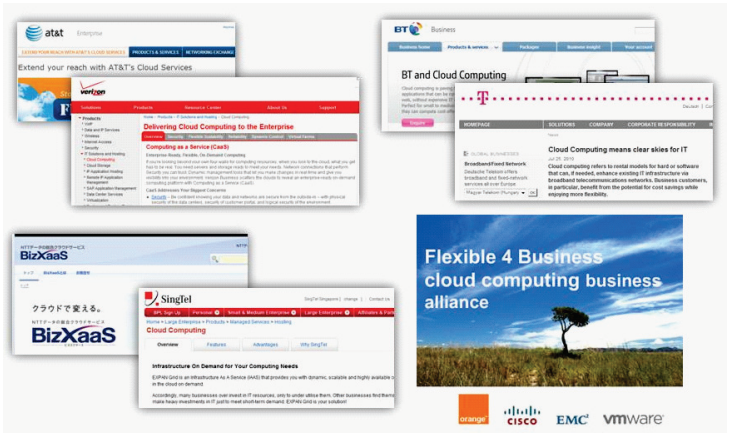


Figure 3 – Example of Recent Cloud Announcements

In general Telco services are very basic so far with Telco Cloud offerings usually added to their existing Hosting Businesses. Most are just entering the market, not wanting to compete on low margin services: 1) Most Telco Offerings are IAAS versions of their hosting businesses as a Phase I. 2) Some Telcos have trials of SMB/Enterprise Offerings, including Telstra, SingTel, NTT Data, Orange and Telefonica. Orange [8] announced recently its value proposition of best in class, simple, secure and easy to pay per use managed services on private cloud with storage, security and unified communications. 3) Some Telcos have new offerings of Storage Clouds like Vodaphone and Orange. 4) Most Telcos are still studying what to do including in terms of PaaS and SaaS, talking to vendors or sending out RFI's. They know they need more sophisticated business models.

In general the value proposition for Telco is therefore: 1) To allows efficiencies & cost savings (e.g. Reduce CAPEX (Elastic architecture fit to needs); Reduce OPEX (Consolidated management); Green (Reduce power consumption)). 2) Opens new business models (e.g. computing / storage services: PaaS, IaaS; SaaS reselling; own SaaS (SDP – like Communications Services, OSS, BSS, BaaS)).

VI. SDP PATTERNS FOR CLOUD COMPUTING

Today, SDP and Cloud Computing are typically different and unrelated stacks, operated by different departments for different projects or business models. Can Cloud Computing further help evolve the core SDP services business of Telcos?

To combine SDP and Cloud Computing, a few different options exist. The most interesting ones as discussed below.

- SDP as resources (or SDP by the Cloud) where SDP exposes its capability in the cloud for applications in the cloud but it is not implemented on the cloud.

 ISBN: 978-1-61208-106-9

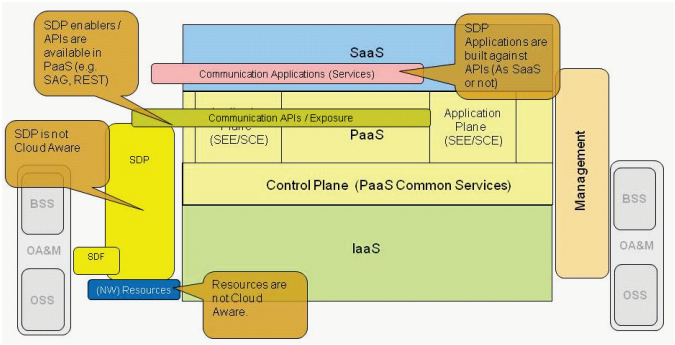


Figure 4 – "SDP by the Cloud".

- With this option, the SDP and its components including enablers as well as underlying resources abstracted via adapters are treated as infrastructure (not cloud based) exposed in IaaS or PaaS via its northbound interfaces with appropriate resource affinity.
- The SDP and underlying resources would be expected to provide (some) scalability support to efficiently support scalable IaaS, PaaS or SaaS here the enablers are used.
- This option enables Telco operators to add SDP (e.g. communications, policies or OSS/BSS) capabilities to a "more conventional computing offering", therefore allowing them to provide a computing on demand offering while differentiating (using their network, SDP, OSS and BSS assets) with respect to other "non Telco" providers.
- This option presents the additional advantage to consist into a relatively straightforward integration of an IaaS or PaaS with possibly an unmodified existing SDP.
- However we do not recommend "porting" the SDP to IaaS differently from the next option (SDP as a PaaS). The present option is rather just a quick way to add communications features to Cloud Computing. In particular this approach does not allow taking advantage of the SDP integration and management via OSS/BSS.
- CMCC OMP (Open Mobile Platform) follows this pattern.

- SDP as a PaaS (or SDP on the Cloud)

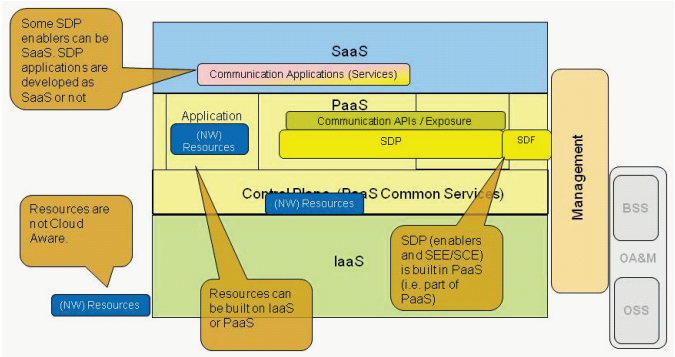


Figure 5 – "SDP on the Cloud".

- A SDP, understood as middleware as previously discussed ([5]), shares its SEE / SCE container(s) with the PaaS (e.g. as a same container or as part of a multiple container PaaS). Resources as defined in SDP (see [1,5]) are preferably implemented in elastic ways (part of IaaS). Resources may of course just be non cloud infrastructure, in which case scalability may be constrained.
- Enablers are part of the PaaS functions (and can therefore be SaaS)
- SDP adapters are built on PaaS to provide resource affinity and support for elastically scalable resources
- Applications can be built on SDP as a PaaS as SaaS or as third party applications (as today) that may be hosted on PaaS or IaaS.
- This option includes the particular cases where the PaaS is just the virtualization of the SDP SEE/SCE running on IaaS.
- This option enables Telco operators to
  - as previously add SDP differentiating capabilities to computing offerings
  - Use Cloud Computing as ways to more efficiently implement its SDP and services e.g. By allowing dynamic allocation of resources based on load demand
  - Provides a path to integrate the Cloud Computing stack with the OSS and BSS, reusing the SDF patterns and capabilities provided by the SDP.
- With such an option, all the services (and enablers) already built on SDP (SEE/SCE) can be easily adapted to the new SDP as a PaaS platform.
- Such an option usually requires support for customizable control plane as mentioned above.

- SDP or Enablers as SaaS (SDP in the Cloud): Enablers are built one way or another as SaaS than can be used by other services.
- The SDP can be any hosted scalable platform implementation, implemented in many ways including the ways described in the previous options or the SDP may not exist and the SDP capabilities and functions are re-implemented on IaaS or PaaS.
  - SDF is not immediately available and it needs to be re-implemented.
- This option may be suitable for new capabilities that can be consisted as enablers or reusable capabilities but are not directly "Telco related" e.g. Search, user profile ... However for existing enablers, it may not be that advantageous considering the previous and following options.
  - This option is especially useful for Telcos to "resell" enablers provided by third parties…

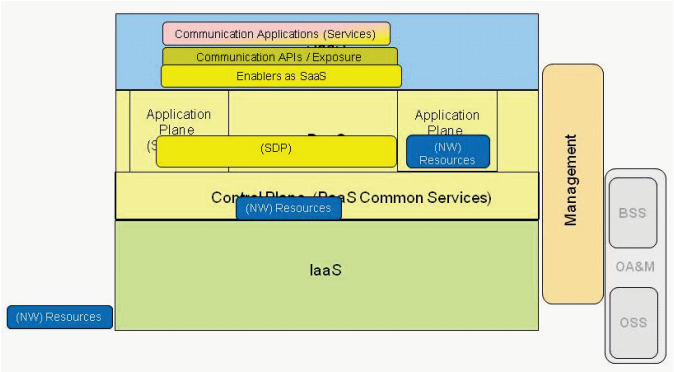


Figure 6 – Example of "Enablers as SaaS".

Again all these patterns can be adopted by generic (i.e., non Telcos) Service Providers or Enterprises building "Communications" offerings.

VII. SDF PATTERNS FOR CLOUD COMPUTING

OSS and BSS integration with SDP provides ways to support:

- OA&M for life cycle management of the SDP and resources as well as applications using or running on it.
- SDF provides support for end to end business process automation across OSS, BS and SDP

With the SDP as a PaaS pattern, the implementation of the PaaS can include appropriate interfaces life cycle management / OA&M of the services, SaaS, PaaS and IaaS (just as for traditional middleware with traditional hardware, OS and applications). Therefore, reusing the SDF and OSS/BSS/SDP integration and adapting the end to end business processes (e.g. to encompass IaaS provisioning on demand and multi-tenancy).

VIII. BLUEPRINT FOR SDP BASED CLOUD COMPUTING SDP

Telcos should target such a combination or roadmap of the above options as appropriate for their objectives and time frame. The end target should be a SDP as a PaaS with enablers as part of

 ISBN: 978-1-61208-106-9

the PaaS or in the Cloud (SaaS) and new related business models: SDP in the Cloud.

With such a SDP as a PaaS (and with our without enablesr as SaaS), Telcos can: 1) More efficiently provide SDP and services (scalable on demand without requiring acquisition or locking of infrastructure to support it). We expect in fact that many target will target PaaS solutions that can target multiple underlying infrastructures like: i) Private Cloud / Next Generation data center, i.e., Private IaaS under the control of the telco, ii) Legacy servers and data centers used as private cloud iii) Public cloud (preferrably as Private Public Cloud) to support Cloud Burst or allow cheap and agile testing environment (while the private assets would be sued for production). Some operators in Europe have started exploring such models. 2) Provide computing services with communications capabilities available to the developers using the service.

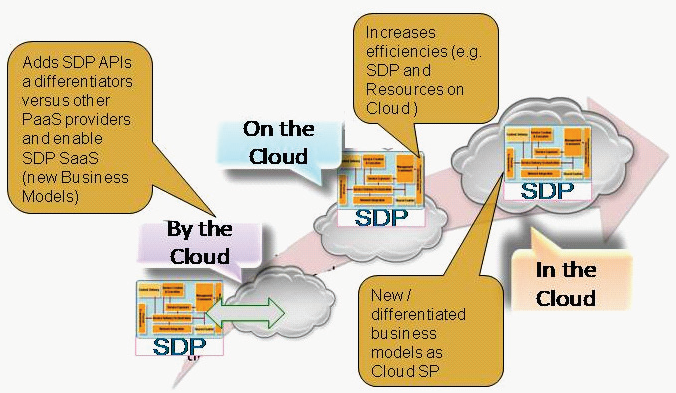


Figure 7 – Evolution of the SDP and the cloud

A. *Examples of Evolution between Options*

As an example of tactical solution, consider for example SaaS SDP communications applications like PBX or Unified communications. They can be built on a JSR 289 (SIP Servlets) / JEE converged container that is virtualized and managed/distributed on an IaaS, while the media servers, conference servers and IVRs are treated first as resources then as scalable resources that can then be built on IaaS. Other "enablers" can be provided first by the cloud. Billing, provisioning and subscription management are driven from the OSS/BSS (via an end to end SOA business processes).

B. *BaaS*

BaaS (Business as a Service) illustrates how Telcos can further differentiate a Cloud Computing offering based on SDP as a PaaS by bundling:

- An on-demand hosted platform to build and execute applications i.e., computing for rent
- On demand backend business applications to build and run a business around the applications:
  - BSS (e.g. CRM, Billing), OSS on demand (as SaaS): i.e., applications for rent
- On demand Customizable pre-build end to end business processes across Platform, OSS, BSS: i.e., business processes for rent.
  - i.e., Pre-built business infrastructure for Rent.

With a BaaS, third parties can rent, develop and run applications and they can also rent a complete backend business infrastructure to run the resulting business as well as all the business processes to automate their business. In other words, besides renting computing and renting say CRM on demand they can rent OSS/BSS on demand and their integration to support selling their applications, campaigning, billing for usage, analyzing the usage, taking new subscriptions, etc.

BaaS is differentiated from conventional IaaS, PaaS and SaaS models. Indeed,

- It is not easy to provide an integrated OSS/BSS/Platform on demand.
- Telco SPs have OSS/BSS assets and end to end Business processes.

IX. FUTURE WORKS

Huawei has numerous SDP deployments with many Telcos currently engaging in Cloud Computing initiatives. Many have reached the point where the Telcos experiments with the patterns and combinations of these patterns described in this paper.

Analysis of lessons learned from such Telco deployments as well as implications on IaaS, PaaS and SaaS and SDP technologies will be provided in upcoming works. Explicit examples today still depend on confidentiality agreements with customers.

X. CONCLUSIONS

This paper provides clarification and recommendations on how to relate SDP and SDF to Cloud Computing. Different patterns that have been discussed are suitable for evolutionary extensions of SDP business model to encompass Cloud Computing.

In particular, we emphasized key value propositions that Cloud Computing can bring to Telcos in the context of SDP and services, including:

- Efficient implementation of SDP and services with always just the resources needed at a given moment.
- New business model to add computing services to the Telco portfolio while being able to successfully differentiate from other computing providers
- BaaS as another differentiated offering that builds on Telco expertise and further facilitate third parties to build business using the services and assets provided by their Cloud Computing service providers.

ACKNOWLEDGMENT

The author wants to thank its colleague at Huawei, especially Dr David Bernstein at the Santa Clara Center for the discussions that have enabled him to reach this view of the field.

REFERENCES

[1] TMF, "Telemanagement Forum", URL: http://www.tmforum.org/

[2] OMA, "OSE, OMA Service Environment", Published by Open Mobile Alliance (OMA), URL: http://www.openmobilealliance.org

[3] 3GPP. IMS, URL: http://www.3gpp.org/ftp/Specs/html-info/22228.htm and http://www.3gpp.org/ftp/Specs/html-info/23228.htm.

[4] S. H. Maes, "Pragmatic Approaches to True Convergence with or without IMS", Network Operations and Management Symposium Workshops, 2008. NOMS Workshops 2008. IEEE.

[5] S. H. Maes, "Service delivery platforms as IT Realization of OMA service environment: service oriented architectures for telecommunications", WCNC 2007, 2007, IEEE

[6] Amazon, "Amazon Web Services (AWS)", URL: http://aws.amazon.com/.

[7] TMForum Insight Services, "Cloud Services, Issues and Opportunities for Service Providers", 2010, TMF[1].

[8] Total Telecom News at URL: http://www.totaltele.com/view.aspx?ID=458926, September 2010.

 ISBN: 978-1-61208-106-9

Financial Business Cloud for High-Frequency Trading

Arden Agopyan
Software Group
IBM Central & Eastern Europe
Istanbul, TURKEY
arden@tr.ibm.com

Emrah Şener
Center for Computational Finance
Özyeğin University
Istanbul, TURKEY
emrah.sener@ozyegin.edu.tr

Ali Beklen
Software Group
IBM Turkey
Istanbul, TURKEY
alibek@tr.ibm.com

Abstract — **This paper defines a new business cloud model to create an efficient high-frequency trading platform. High-frequency trading systems, built to analyze trends in tick-by-tick financial data and thus to inform buying and selling decisions, imply speed and computing power. They also require high availability and scalability of back-end systems which require high cost investments. The defined model uses cloud computing architecture to fulfill these requirements, boosting availability and scalability while reducing costs and raising profitability. It incorporates data collection, analytics, trading, and risk management modules in the same cloud, all of which are the main components of a high-frequency trading platform.**

Keywords — high-frequency trading; cloud computing; financial business cloud;

I. INTRODUCTION

Financial markets are broad and complex systems in which market players interact with each other to determine the prices of different assets.

Advances and innovations in computer technologies have changed the nature of trading in financial markets. As a result of these innovations, transmission and execution of orders are now faster than ever, while the holding periods required for investments are compressed. For this reason a new investment discipline, high-frequency trading, was born [1].

In very broad terms, high-frequency trading refers to analyzing trends in tick-by-tick data and basing buying and selling decisions on it.

Exchanges supporting high-speed low-latency information exchange have facilitated the emergence of high-frequency trading in the markets. In 2009, in the United States, high-frequency equity trading was 61% of equity share volume and generated $8 billion per year [2]. Again in the United States, high-frequency trading also accounted for up to 40% of trading volume in futures, up to 20% in options, and 10% in foreign exchange [3]. It has already become popular in Europe and is also manifesting itself in some emerging markets, like Latin America and Brazil [3]. It is estimated that about 30% of Japanese equity trading is high-frequency [3]. This compares with up to 10% in all of Asia, up to 10% in Brazil, about 20% in Canada, and up to 40% in Europe [3].

High-frequency trading platforms incorporate trading, data collection, analytics, and run-time risk management modules to create systems which search for signals in markets, such as price changes and movements in rates. This helps to spot trends before other investors can blink. Then finally orders and strategies are executed or changed within milliseconds on the exchanges. The trading module hosts trading algorithms built on top of the statistical models and executes orders on electronic execution platforms like exchanges. The data collection module collects tick-by-tick data from data providers and feeds trading and analytics modules. This data can also be exported to external data analysis tools. The analytics module is used to analyze historical financial data, to generate automated reports and to help creating new trading algorithms. Finally, the run-time risk management module is responsible for maintaining the whole system within pre-specified behavioral and profit and loss boundaries. These modules can be accessed via web and rich mobile applications which enhance management capabilities and increase the speed of user interactivity and control.

High-frequency trading systems imply speed, as high-frequency trades are done in milliseconds, and also require high availability and readiness to trade at anytime. The speed of execution is secured by powerful hardware and co-location of the systems with the electronic execution platforms to minimize the network latency [1, 4]. High availability is achieved by adding more resources to the system and by clustering the datacenters. All of these necessitate high cost investments.

Cloud computing refers to both the applications delivered as services with Software as a Service (SaaS) model over the Internet, and the hardware and systems software in the datacenters that provide those services [5]. A cloud is the ensemble of applications delivered as services and datacenter hardware, software and networking.

From the cloud user and consumer perspective, in the cloud, computing resources are available on demand from anywhere via the Internet and are capable of scaling up or down with near instant availability. This eliminates the need for forward planning forecasts for new resources [6]. Users can pay for use of computing resources as needed (e.g., processors by the hour and storage by the day) and

 ISBN: 978-1-61208-106-9

release them as needed, thereby rewarding conservation by letting machines and storage go when they are no longer useful [5]. The cost impact of over-provisioning and under-provisioning is eliminated [6] and consumers no longer need to invest heavily or encounter difficulties in building and maintaining complex IT infrastructures [7]. Cost elements like power, cooling, and datacenter hardware and software are eliminated, as well as labor and operations costs associated with these. Using computing as a utility [6] with infinite and near instant availability and low entry costs gives enterprises the opportunity to concentrate on business rather than IT in order to enter and exploit new markets. There is also no cost for unexpectedly scaling down (disposing of temporarily underutilized equipment), for example due to a business slowdown [5]. In our world, where estimates of server utilization in datacenters range from 5% to 20% [5], elastic provisioning to scale up and down to actual demand creates a new way for enterprises to scale their IT to enable business to expand [6].

In cloud computing, business process as a service is a new model for sharing best practices and business processes among cloud clients and partners in the value chain [8]. A business cloud covers all scenarios of business process as a service in the cloud computing environment [8].

This paper presents a financial business cloud model for high-frequency trading to create an efficient trading platform and IT infrastructure using cloud computing architecture for financial institutions. In this model, trading, data collection, analytics, and run-time risk management modules are deployed to the cloud. An Enterprise Service Bus, a standard-based integration platform [9], integrates these modules and handles routing, data transformations, mediations and messaging between them. Cloud Manager is responsible for essential tasks like policy management, account management, authorization & access, security, application management, scheduling, routing, monitoring, auditing, billing and metering [8]. It exposes modules as high availability financial cloud services accessible from anywhere in the world via the Internet. The whole cloud is co-located in datacenters close to the electronic execution platforms to avoid data movement costs and network latency, and to assure the speed of execution [4, 5].

Cloud computing is a unique opportunity for batch-processing and analytics jobs which analyze terabytes of data and take hours to finish, as well as automated tasks responsible for responding as quickly as possible to real-time information [5]. As these are essential jobs in high-frequency trading operations, and require high computing power, high-frequency trading platforms are ideal candidates for cloud computing.

Total cost of ownership can be reduced by using high-frequency trading platforms as financial business clouds instead of deploying capital intensive on-premise infrastructure. Adopting this model reduces the IT dependence of high-frequency trading while increasing profitability. Existing systems can be designed to exist in a cloud, as portability can be achieved while moving to cloud environments [10].

Cloud computing gives financial institutions the opportunity to outsource their IT infrastructure and operations, and to concentrate on business rather than IT. It also helps to reduce their operational risk and risk management costs because availability and service delivery are assured by cloud providers via Service Level Agreements (SLAs) [7]. Cloud computing has a big future for high-frequency trading clients, and can be used increasingly to allow firms to implement strategies that previously might have been considered too short-term to justify implementation [11].

Section 2 of this paper, presents work related to this subject. Section 3 discusses why high-frequency trading requires the adoption of cloud computing as Information Technology (IT) infrastructure. This section also includes the reference component architecture of a contemporary on-premise high-frequency trading platform. Section 4 reveals the proposed model with a case study which helped to determine the requirements of the model. Section 5 presents the conclusion and future work.

II. RELATED WORK

There are many published studies to assist in understanding high-frequency trading and cloud computing individually. Irene Aldridge published a book exploring various aspects of high-frequency trading [1], and references [5, 6, 7, 8] are valuable studies on cloud computing. Regarding financial cloud applications, V. Chang, G. Wills and D. De Roure proposed the Financial Cloud Framework [10]. This study demonstrates how portability, speed, accuracy and reliability can be achieved while moving financial modeling from desktop to cloud environments.

This study proposes a financial business cloud model and addresses high-frequency trading. It proposes cloud reference architecture for efficient high-frequency operations.

III. HIGH-FREQUENCY TRADING AND CLOUD COMPUTING

This section examines why high-frequency trading requires the adoption of cloud computing as IT infrastructure. The reference component architecture of a contemporary on-premise high-frequency trading platform is also presented.

A. High-Frequency Trading

In time, masters of physics and statistics, quants, gave birth to quantitative trading. This is a new trading style using innovative and advanced mathematical trading models which make portfolio allocation decisions based on scientific principles. The objective of high-frequency trading is to run the quant model (the model developed after quantitative analysis) faster, and to capture the gain from the market, as high-frequency generation of orders leaves very little time for traders to make subjective non-quantitative decisions and input them into the system.

 ISBN: 978-1-61208-106-9

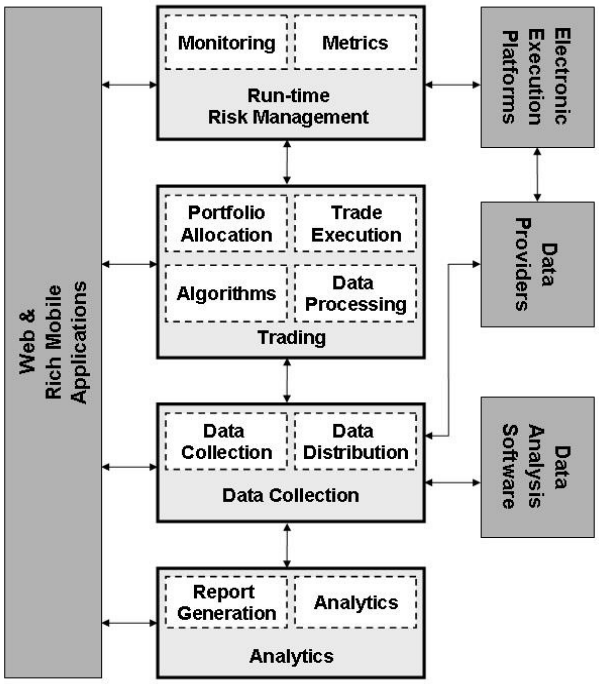


Figure 1. Reference component architecture of a contemporary on-premise high-frequency trading platform.

Many high-frequency traders collect tiny gains, often measured in pennies, on short-term market gyrations [12]. They look for temporary "inefficiencies" in the market and trade in ways that can make them money before the brief distortions go away [12].

The need for speed, to make and execute trading decisions and strategies, requires investment in fast computers. These strategies are established by designing algorithms including generation of high-frequency trading signals and optimization of trading execution decisions. The need to be ready to trade at anytime requires high availability of the trading and execution systems. This high availability is assured by adding more resources to the system and by clustering the datacenters. With all of these aspects, high-frequency trading operations are IT dependent.

This IT dependence of high-frequency trading generates two drawbacks from a cost perspective:

- Profitability: Trading itself already entails a transaction cost, and high-frequency trading generates a large number of transactions, leading to exorbitant trading costs. As high-frequency traders look for tiny gains, the combination of trading and IT infrastructure costs reduces profitability.
- Lead time to deploy trading algorithms and strategies: Implementing high-frequency trading platforms to deploy algorithms and strategies created by quants and traders requires experienced IT labor and this adds another layer to the operation, costing time and money.

B. Contemporary High-Frequency Trading Platforms

Contemporary high-frequency trading platforms incorporate trading, data collection, analytics, and run-time risk management modules. They may also be accessed via web and rich mobile applications to provide user control and enhanced management capabilities.

Figure 1 shows the reference component architecture of a contemporary on-premise high-frequency trading platform. In this architecture:

- The trading module incorporates optimal execution algorithms to achieve the best execution within a given time interval, and the sizing of orders into optimal lots while scanning multiple public and private marketplaces simultaneously. These algorithms are generally academic researches and proprietary extensions which are coded and embedded into the software. This module accepts and processes data from data providers via the data collection module and real-time data coming from exchanges. It generates portfolio allocation and trade signals, and records profit and loss while automating trading operations.
- The data collection module is responsible for collecting real-time and historical financial data coming from data providers. High-frequency financial data are observations on financial variables taken daily, or on a finer time scale, and this time stamped transaction-by-transaction data is called tick-by-tick data [11]. Data providers (or aggregators) are companies who generally provide 24-hour financial news and information including this high-frequency real-time and historical price data, financial data, trading news and analyst coverage, as well as general news. Collected tick-by-tick data and financial news in machine readable format are distributed to trading and analytics modules to feed trading algorithms, to support decision making processes, and to generate reports. This data can be exported to external data analysis software to be used in algorithmic research.
- The analytics module is responsible for automated report generation from historical financial data as well as providing multi-dimensional analytics.
- The run-time risk management module ensures that the system stays within pre-specified behavioral and profit and loss bounds using pre-defined metrics. Such applications may also be known as system-monitoring and fault-tolerance software [1].
- The electronic execution platform is the exchange or market facilitating electronic trading (preferably in high-speed and low-latency) which is a must for high-frequency trading operations. Platform independent high-frequency systems can connect to multiple electronic execution platforms. Intermediary languages like Financial Information

 ISBN: 978-1-61208-106-9

eXchange (FIX), a special sequence of codes optimized for the exchange of financial trading data, helps organizations to change the trading routing from one executing platform to another, or to several platforms simultaneously [13].

- Web and rich mobile applications are channels developed to enhance management capabilities, and increase the speed of user interactivity and control. They may also incorporate modules under the same interface to create a single point of control.

Modules can be developed in-house, or alternatively proprietary software sold by major software vendors can be used. Modules are deployed on-premise following high investments in expensive datacenters including hardware, software and network connectivity [5]. Generally, each module is deployed on-premise to separate hardware with very low or no virtualization. They interact with each other independently with different communication protocols and data types. Development, deployment, operation and maintenance of these systems require experienced IT labor which is expensive and drives costs upwards.

C. Cloud Computing as Infrastructure for High-Frequency Trading

The adoption of cloud computing as infrastructure for high-frequency trading addresses the IT dependency of high-frequency trading platforms as follows:

- Investing in building and maintaining complex IT infrastructure is no longer necessary. Computing resources are billed on a usage basis.
- Computing resources are infinitely available on demand from anywhere via the Internet.
- The cloud provider is responsible for maintaining and operating the IT infrastructure.

Most of the tasks in high-frequency trading operations are automated based on algorithms. The whole system is responsible for responding as quickly as possible to real-time information coming from markets. Cloud computing provides the availability, speed and computing power required for these automated operations.

High-frequency trading operations include batch-processing and analytics jobs requiring high computing power. Cloud computing provides a unique opportunity in this regard [5].

Total cost of ownership can be reduced by adopting cloud computing as a high-frequency trading infrastructure instead of deploying capital intensive on-premise infrastructure. Buyers can move from a capital expenditure (CAPEX) model to an operational expenditure (OPEX) one by purchasing the use of the service, rather than having to own and manage the assets of that service [6]. Adopting this model reduces the IT dependency of high-frequency trading while increasing profitability.

Nowadays, trading firms and hedge funds are already outsourcing their accounting and back-office operations. Cloud computing gives financial institutions the opportunity to outsource their IT infrastructure and operations, and concentrate on business rather than IT. It also helps to reduce their operational risk and risk management costs because availability and service delivery are assured by cloud providers via SLAs [7]. As high-frequency trading operations are already running in many countries, this model will facilitate the entry of other participants to the market at a low entry cost. Cloud computing has a big future for high-frequency trading clients and can be used increasingly to allow firms to implement strategies that previously might have been considered too short-term to justify implementation [11].

IV. FINANCIAL BUSINESS CLOUD FOR HIGH-FREQUENCY TRADING (FBC-HFT)

This section presents the Financial Business Cloud for High-Frequency Trading (FBC-HFT) to create an efficient trading platform and IT infrastructure for financial institutions using cloud computing architecture. A case study which helped determine the requirements of FBC-HFT is also exposed in this section.

A. A Case Study to Determine Requirements of FBC-HFT

Implementation of a high-frequency trading platform consists of many components such as identified statistical models, coded algorithms using these models to analyze and clean the tick data, installations of hardware and software, and connections with exchanges and data providers as well as whole risk management structure of the platform. The objective of this case study is to analyze historical high-frequency foreign exchange data extracted from Bloomberg [14] to determine the main requirements of FBC-HFT for the data analysis phase, which is the most critical part of the operation. Implementation of other components required to build a complete high-frequency trading platform is subject to future work.

High-frequency TRY/USD currency data between September 12th 2009 and March 27th 2010 are used for this application.

Extracted data are already ordered, filtered and cleaned by Bloomberg, so no further cleansing required (data cleansing may be required for different types of data from different providers). Business week restrictions are not applied to the data as the analysis is not region specific.

Intervals of one minute, five minutes, ten minutes, thirty minutes, one hour, six hours, twelve hours and daily tick data for closing trade prices are extracted and the time series created from these data sets have following fields:

- A timestamp (ex: 12.09.2009 09:00)
- Last trade price (ex: 1.484)
- A financial security identification code (ex: TRYUSD)

Last trade prices are not meaningful in isolation, so calculation of return series (R) from these data sets is required. Returns are expressed as percentages and are calculated using the following formula:

$$R = \ln\left(\frac{P_t}{P_{t-1}}\right) \quad (1)$$

 ISBN: 978-1-61208-106-9

All analysis and experiments are done using these calculated return series data sets.

All experiments include the following analysis for one minute, five minutes, ten minutes, thirty minutes, one hour, six hours, twelve hours and daily tick data:

- Basic descriptive statistics (including mean, standard deviation, sample variance, kurtosis and skewness)
- Histogram for the frequency of return percentages (to graphically interpret skewness and kurtosis)

A graph of changes in kurtosis of all time series is generated to show kurtosis' behavior. In the case of one minute and daily tick data, Augmented Dickey-Fuller test for unit root testing (test for stationarity), and correlograms to check the auto-correlation function, are calculated.

Generated basic descriptive statistics and histograms for one minute data are shown in Table 1 and in Figure 3 respectively.

This case study presents an essential part of high-frequency trading operations which includes:

- Extraction of the data from data providers,
- Conversion and manipulation of the data for different analysis software and tools,
- Routing the data to the tools.
- Analyzing the data for high-frequency characteristics and decision-making based on this analysis.

Regarding this case study, the following outputs are observed:

- Development of data conversions and transformations is time consuming and hampers the implementation.
- There is a need for integration between different tools and systems.
- Analyzing high-frequency data is computing power intensive.

These outputs show that the adoption of cloud computing can address the computing power need. An Enterprise Service Bus (ESB), is a standards-based integration platform combining messaging, web services, data transformation and intelligent routing in a highly distributed, event driven Service Oriented Architecture [9], that can facilitate the development of data transformation and the integration of different systems.

B. The Model

The proposed reference model in this paper incorporates high-frequency trading modules in short running; routing, data and protocol conversion based processes and reveals them as a business cloud.

Figure 2 shows the reference component architecture of the proposed Financial Business Cloud for High-Frequency Trading.

In this architecture, trading, data collection, analytics, and run-time risk management modules are deployed to the cloud. Existing systems can be designed to exist in a cloud as portability can be secured while moving to cloud environments [10]. Their functionalities and roles in the operation are the same as in contemporary high-frequency trading platforms. However, the integration of these modules, routing and data, and protocol conversions between them, are now handled with an ESB. Modules provide standardized interfaces to be accessed and managed in the cloud.

Cloud Manager (CM) is the common management system which also manages request and response flows in the cloud. CM is directly connected to electronic execution platforms and data providers. Modules which need interaction with electronic execution platforms and data providers use CM to access outside the cloud. All routing, data and protocol transformations, mediations and messaging between modules and CM are done via ESB. This provides flexibility and standardized integration of the system components.

CM provides web and rich mobile application channels as single points of control for the cloud, boosting the speed of user interactivity and control. Data for external data analysis software can be exported via Cloud Manager.

CM is also responsible for cloud specific management tasks:

- Account management for cloud users and consumers.
- Authorization and access control of users for modules and resources.

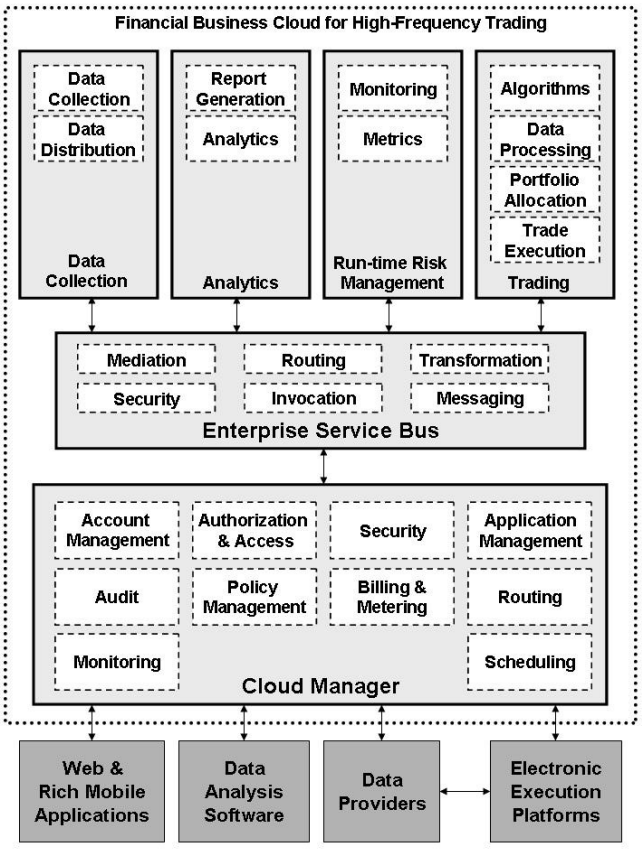


Figure 2. Reference component architecture of Financial Business Cloud for High-Frequency Trading.

- Scheduling of jobs and tasks as well as selecting and provisioning suitable resources in the cloud.
- Routing of incoming requests from outside the cloud to the ESB to run associated processes, and vice versa.
- Application management for deployed applications (modules) including application specific configurations.
- Policy management for cloud resources and configuration of SLAs guaranteeing service availability and delivery.
- Monitoring of the entire cloud including users, tasks, processes, modules and resources.
- Security of the cloud.
- Providing audit records of the cloud.
- Metering, usage-based billing and billing management.

The whole cloud is co-located in datacenters close to the electronic execution platforms to avoid data movement costs and network latency, and to assure speed of execution [4, 5].

The Financial Business Cloud for High-Frequency Trading is a model to adopt cloud computing as an IT infrastructure for financial institutions running high-frequency operations. It brings the benefits of cloud computing to high-frequency trading and addresses business specific issues explained in the previous sections.

V. CONCLUSION AND FUTURE WORK

This paper presents a new business cloud model to create an efficient high-frequency trading platform. Current drawbacks and needs of high-frequency trading are addressed by the proposed reference model.

Future research and work on this study will implement a complete real life scenario of this reference model to test performance benefits and the efficiency of the system, from both cloud consumer and cloud provider perspectives. The implementation and development of each component of the proposed model, and the development of security and management approaches are also subject to future work.

REFERENCES

[1] I. Aldridge, High-Frequency Trading. New Jersey: Wiley, 2010.

[2] L. Tabb, R. Iati, and A. Sussman, “US Equity High Frequency Trading: Strategies, Sizing and Market Structure”, 2009, Report by TABB Group.

[3] Internet: High-frequency trading surges across the globe. Available on WWW at URL: http://www.dnaindia.com/money/report_high-frequency-trading-surges-across-the-globe_1319167 (Last access date: June 2010).

[4] Internet: High-frequency trading. Available on WWW at URL:http://www.vimeo.com/6056298 (Last access date: June 2010).

[5] M. Armbrust et al., “Above the Clouds: A Berkeley View of Cloud Computing”, Technical Report, 2009. Electrical Engineering and Computer Sciences, University of California at Berkeley, February 2009.

[6] M. Skilton et al., “Building Return on Investment from Cloud Computing”, White Paper, The Open Group, April 2010.

[7] R. Buyyaa, C. S. Yeoa, S. Venugopala, 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, December 2008.

[8] L. Zhang and Q. Zhou, “CCOA: Cloud Computing Open Architecture”, IEEE International Conference on Web Services, 2009

[9] D. A. Chappell, Enterprise Service Bus, O'Reilly Media, June 2004, p.1.

[10] V. Chang, G. Wills, and D. De Roure, “Towards Financial Cloud Framework - Modelling and Benchmarking of Financial Assets in Public and Private Clouds”, University of Southampton.

[11] Internet: New wave of high-frequency traders to target European markets. Available on WWW at URL:http://www.thetradenews.com/node/4395 (Last access date: June 2010).

[12] Internet: What's Behind High-Frequency Trading. Available on WWW at URL: http://online.wsj.com/article/SB124908601669298293.html (Last access date: June 2010).

[13] Internet: What is FIX? Available on WWW at URL: http://www.fixprotocol.org/what-is-fix.shtml (Last access date: June 2010).

[14] Internet: Bloomberg - Business & Financial News, Breaking News Headlines. Available on WWW at URL: http://www.bloomberg.com (Last access date: August 2010).

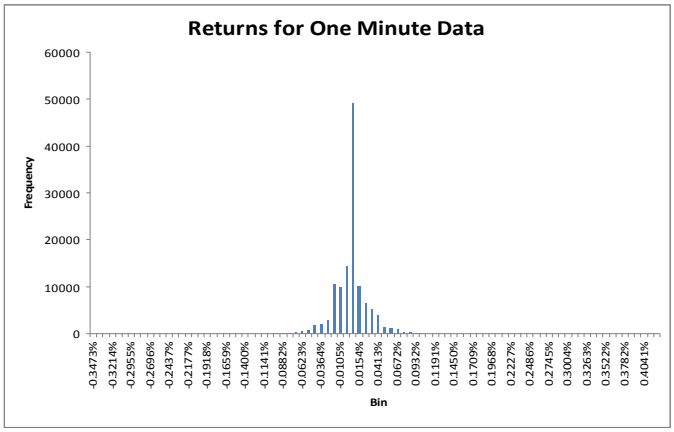


Figure 3. Histogram of returns for one minute data.

TABLE I. BASIC DESCRIPTIVE STATISTICS FOR ONE MINUTE DATA

Mean	4.73023E-07
Standard Error	8.1594E-07
Median	0
Standard Deviation	0.000287753
Sample Variance	8.28017E-08
Kurtosis	265.2351796
Skewness	-2.318022932
Range	0.030399821
Minimum	-0.019277394
Maximum	0.011122428
Sum	0.058830806
Count	124372
Largest(1)	0.011122428
Smallest(1)	-0.019277394
Confidence Level(95,0%)	1.59923E-06

Cloud Computing for Online Visualization of GIS Applications in Ubiquitous City

Jong Won Park, Yong Woo LEE, Chang Ho Yun, Hyun Kyu Park, Seo Il Chang, Im Pyoung LEE, Hae Sun Jung*
The Ubiquitous (Smart) City Consortium,
The University of Seoul, *Korea University
{comics77, ywlee, touch011, ajnick31, schng, iplee}@uos.ac.kr, holylife7@hotmail.com

***Abstract*— Cloud computing can be used to generate the 3D noise maps in ubiquitous cities. Here in this paper, we present our cloud computing approach, its performance and a performance comparison for it. The 3D image processing with GIS data requires great amount of computational resource because of complex and large amount of spatial information. The cloud computing can solve the problem with an easy and transparent way. We use Hadoop which is a framework that includes the HDFS (Hadoop Distributed File System) and MapReduce as cloud computing methodology to do massively parallel processing of 3D GIS data. We found the computing time is vastly reduced with a cluster of computing nodes. We also present the performance comparison when we use MPI instead of MapReduce and Hadoop.**

Keywords- cloud computing; the noise map; GIS; Hadoop; MapReduce; MPI.

I. INTRODUCTION

In the 1990s, the noise map was presented to develop the environmental policy to reduce the noise in cities. Afterward, in 2002, Directive 2002/49 relating to the assessment and management of environmental noise was adopted by the European Parliament and Council for the developments of the long-term noise policy. The European Environmental Noise Directive (2002/49/EC) is one of the European Community's policies which have the goal to avoid, prevent, and decrease their displeasure and harmful effect caused by environmental noise exposure [1].

We find that immediately after the standard about the noise map was adopted by the EC, European Initiatives on the research of the noise map have been activated. The noise map combines noise information with GIS map. It requires a large amount of computing power and cannot be timely done with personal computers. In the reason, the noise map is usually made offline mode for long time and not in three-dimension but in two-dimension. However, current cities have high-rising buildings and we need to show the noise difference on each floor. In consequence, it is important to generate the 3D noise map [2][3]. The 3D image processing with GIS data should deal with complex and large amount of spatial information and requires great amount of computational resource.

In this paper, we present our approach to solve the problem in two ways and compare the performance. One way is to use MapReduce [4] with Hadoop system [5] and the other way is to use MPI.

The structure of this paper is as follows. In Section 2, we introduce, compare and analyze the state-of-the art works related to our research. In Section 3, we explain the steps of noise map. In Section 4, we describe our cloud computing approach to do it. In Section 5, we give performance evaluation. Finally, we conclude and explain the future work in Section 6.

II. RELATED WORK

EU has been actively researched noise map. Table 1 shows EU countries and their participating cities in the research [6]. Figure 1 and Figure 2 show some of their research results, that is, two noise maps in two-dimension. They do not produce online noise maps but makes the noise maps in offline mode and do not use cloud computing. The research on the 3D-noise map is an arising topic and not found except our work.

TABLE I. EUROPEAN UNION (EU) COUNTRIES AND THEIR CITIES MAKING THE NOISE MAP

Country	City
United Kingdom	London, Birmingham
Germany	Berlin
France	Paris
Netherlands	Amsterdam
Czech	Prague
Italy	Bologna
Switzerland	Geneva
Austria	Vienna
Sweden	Stockholm
Finland	Helsinki
Belgium	Brussels

 ISBN: 978-1-61208-106-9

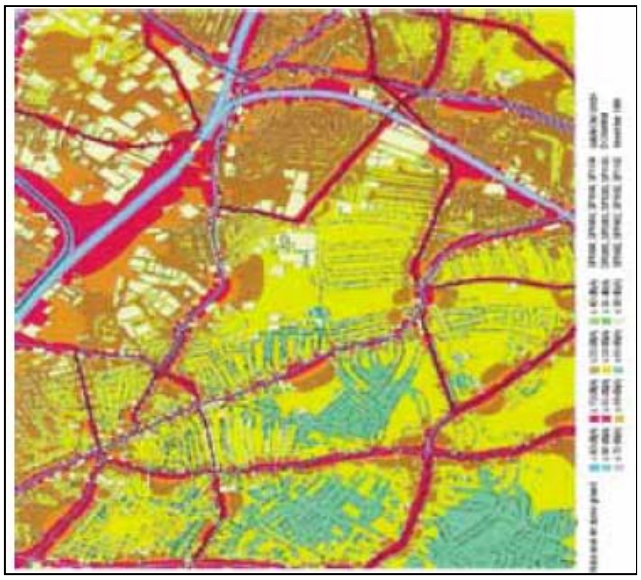


Figure 1. A noise map of Birmingham, U.K.

Figure 2. A noise map of Amsterdam, Netherlands

III. How to Generate the Noise Map?

The noise map is currently made in two dimensions. However, in this research, we are interested in the three dimension noise map. In this paper, we explain our way to generate the three dimension noise map, not the two dimension noise map. In order to make a 3D noise map, the following three-step-process is needed: 1) Making a noise database. 2) Generating the 3D city model. 3) Integrating the noise values with the 3D city model.

A. *Making a Noise Database*

In our ubiquitous cities, the noise data are collected through ubiquitous sensor network from remote sensors and sent to the database. We can also use an interpolation approach to make the database. That is, we measure the noise at important areas and use a noise prediction model to predict noise values at unmeasured areas using the measured data at the area nearby the unmeasured area [7].

B. *Generatinga 3D City Model*

The generation of 3D city model includes the terrain modeling as shown in Figure 3 and the building modeling [8] as shown in Figure 4. It needs big computing power because generation of the 3D building model is very complicated. To solve the problem, we use the cloud computing [9].

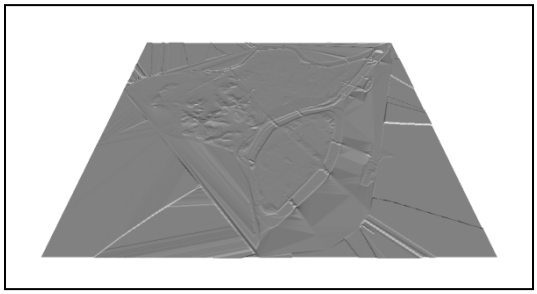


Figure 3. A digital elevation model

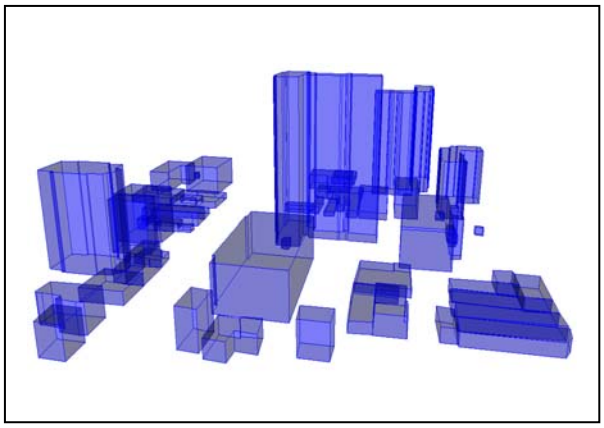


Figure 4. Building models

C. *Integrating the Noise Values with the 3D City Model*

Now, we integrate the noise values with the 3D city model. Because the data of 3D city model is very large, converting each noise value into RGB value and mapping the RGB value onto the texture file of the 3D city model requires a large amount of computing power. To solve the problem, again, we use the cloud computing. Thus we can reduce the running time to the level of online processing. Figure 5 shows a sample digital map of an experimental area.

 ISBN: 978-1-61208-106-9

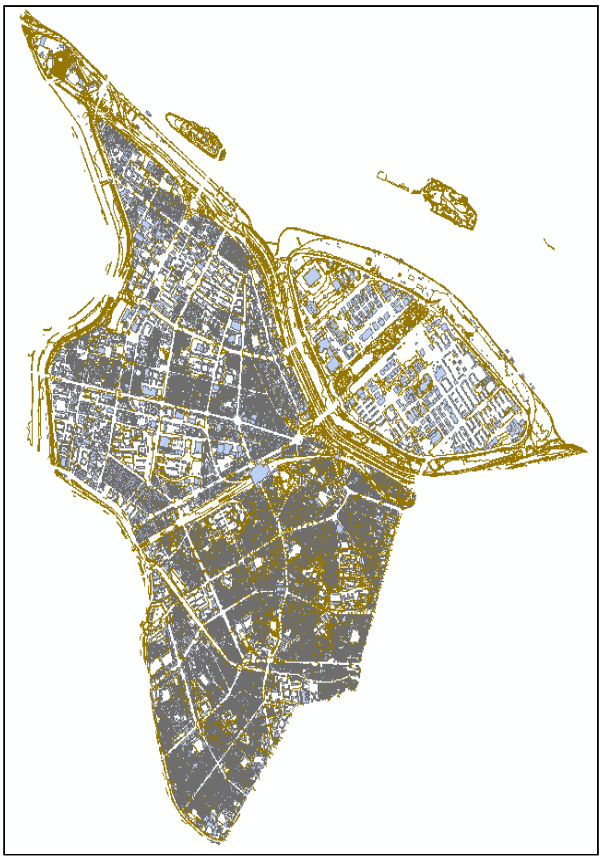


Figure 5. The digital map of Yeongdeungpogu Disrict, Seoul, Korea

IV. THE CLOUD COMPUTING

The cloud computing process to make the 3D noise map is shown in Figure 6. To process 3D data, we employ the ways that the data of the digital map are divided into grid cell units. The data of the digital map make a huge file so we use the MapReduce with Hadoop that is one of cloud computing technologies to do massively distributed and parallel processing. Distributed and parallel programming greets the new trends due to the cloud technologies such as Hadoop, an open source Java framework. It consists of Hadoop Distributed File System (HDFS) and MapReduce. HDFS uses a scheme of replication to ensure that the stored files are always kept intact in separate places of a Hadoop cluster. It enables us to solve a large scale of data intensive problems.

We divide the data of the 3D GIS images into the unit of grid cell for the MapReduce processing and later integrate the result since the MapReduce uses Single-Program Multiple-Data (SPMD) methodology [10]. As shown in Figure 7, we used MapReduce to make the 3D city model and the 3D noise map and the generated 3D city model is reused as an input to the 3D noise map.

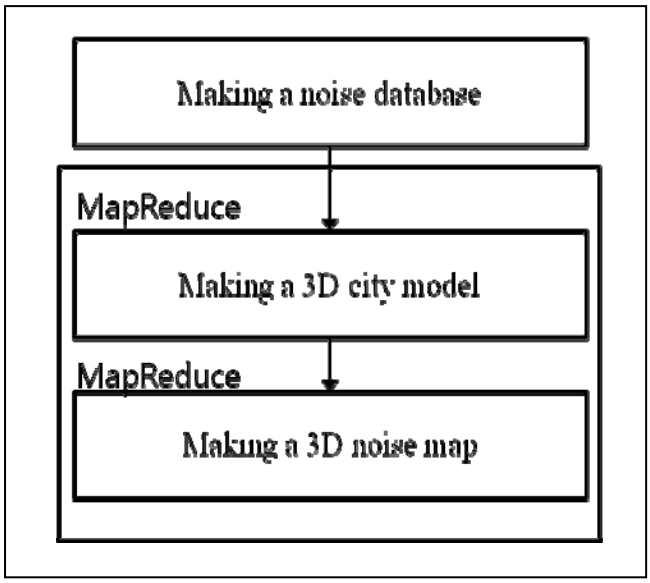


Figure 6. The cloud computing process to make the 3D noise map

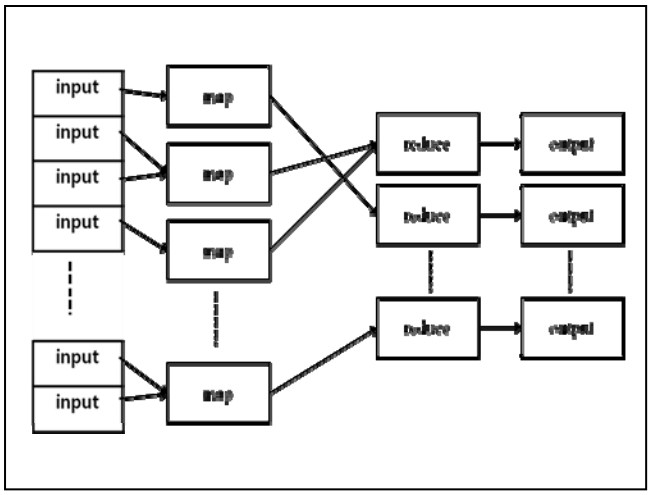


Figure 7. The MapReduce execution

A. *Cloud Computing to Make a 3D City Model*

Here, we explain how we do cloud computing with MapReduce to make the 3D city model. We use two kinds of map functions: map_1 and map_2. Map_1 plays a role of making a Digital Elevation Model (DEM), also called as a Digital Terrain Model (DTM), which has the topology information and height information of ground surface for the 3D city model. Map_2 plays a role of making a building model which has the object topology information and the height information as shown in Figure 4. Reduce integrates the DEM and the building model: this process is called as reduce_1. The output of the reduce_1 is a 3D city model. Table 2 explains the three functions: map_1, map_2 and reduce_1.

 ISBN: 978-1-61208-106-9

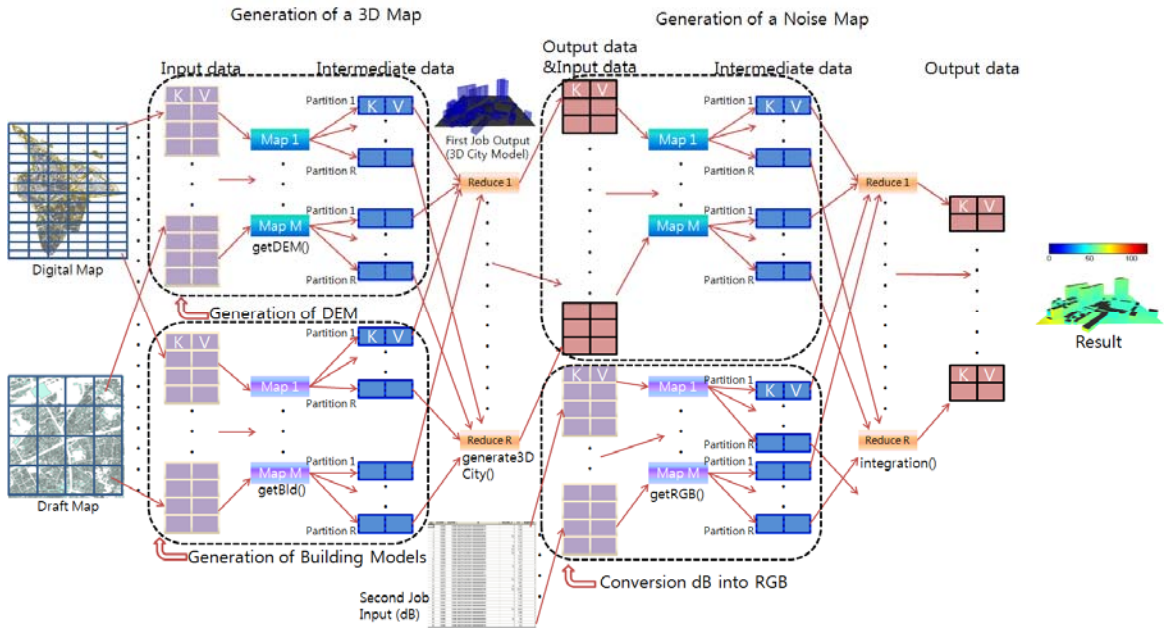


Figure 8. The process of MapReduce function in visualization of the 3D noise map

TABLE II. THE TASKS TO MAKE A 3D CITY MODEL

Function	Task	Key-Value Pair
Map_1	To make a DEM.	<<Sub-area ID, x, y coordinates>,<z coordinates, topography ID >>
Map_2	To make a building model.	<<Sub-area ID, x, y coordinates>,<z coordinates, building ID >>
Reduce_1	To integrate the DEM and the building model.	<<Sub-area ID, x, y coordinates>, <z coordinates, value of 3d city model>>

To make the building model, the getBld() of the map_2 extracts the coordinates of 2D building boundary from the digital map and extracts the z value of the building from the draft map by establishing the correspondence between the building in the digital map and it in the draft map.

We divided the test area into a number of sub-areas and assigned an ID to each of them. When the test area is processed in the map function, the coordinates of the specific area is assigned with a sub-area ID. Therefore, the key value becomes <sub-area-ID, x, y-coordinate>.

The outputs of "map_1" and "map_2" are sorted and grouped according to the ID by the partitioner. The outputs of the partitioner become the input of "reduce_1". It means that the key-value pairs of the DEM and the building model that are sorted and grouped become the inputs of "reduce_1". "Reduce_1" calls and process the generate3DCity() function to generate the 3D city model. Each "reduce_1" task is matched to each sub-area and therefore the number of the "reduce_1" task is same as the number of the sub-areas. The output of "reduce_1" will be used as the input of "map_4" in the next step.

B. *Cloud Computing to Making a Noise Map*

Here, we explain how we combine the 3D city model with the noise information to generate the 3D noise map. We use two kinds of map functions: map_3 and map_4. Map_3 takes the noise information of buildings as the inputs of reduce_2. As the output, we take the key-value pair of <<building ID, x, y coordinates> and <z coordinates, value of noise level>>. Map_4 transfers the result of reduce_1 to make the 3D noise map. Table 3 explains the three functions: map_3, map_4 and reduce_2.

TABLE III. THE TASKS TO MAKE A NOISE MAP

Function	Task	Key-Value Pair
Map_3	To take the noise information of buildings as the inputs of reduce_2.	<<building ID, x, y coordinates>, <z coordinates, value of noise level>>
Map_4	To transfer the result of reduce_1 to reduce_2.	<<Sub-area ID, x, y coordinates>, <z coordinates, value of 3d city model >>
Reduce_2	To integrate the 3D city model and noise information.	A noise map.

The input of map_3 has coordinates, building ID and noise value to make a noise map. Because it has a coordinates, the noise value can be matched to 3D City

model. As a key of map_3, we use a building_ID to distinguish each building.

The outputs of "map_1" and "map_2" are sorted and grouped according to the ID by the partitioner. The outputs of the partitioner become the input of "reduce_2". Reduce_2 plays a role of visualizing the RGB by combining the inputs with 3D city model. We divide noise level distribution of the test area into sub areas and find RGB color index using getRGB() function. When we convert it into color index, we use the equation as shown in Eq. (1) [3]:

$$NC = \frac{(C-1)*(N-N_{max})}{N_{max}-N_{min}} \quad (1)$$

In Eq. (1), NC is color index, Nmax is the maximum value of the noise pollution level, Nmin is its minimum value, C is the total number of colors and N is the noise level on each grid.

After noise level is converted into RGB values, the following steps are performed to generate the noise map. First, the group of the points corresponding to each wall facet is classified according to the proximity of each point to the facet. Second, the RGB of the classified points are then interpolated into a grid using the same encoding scheme presented as Eq. (1). Now, we can get facet image files and merge the files into one file so as to generate a 3D noise map. The final result of the merged file is the 3D noise map as shown in Figure 9.

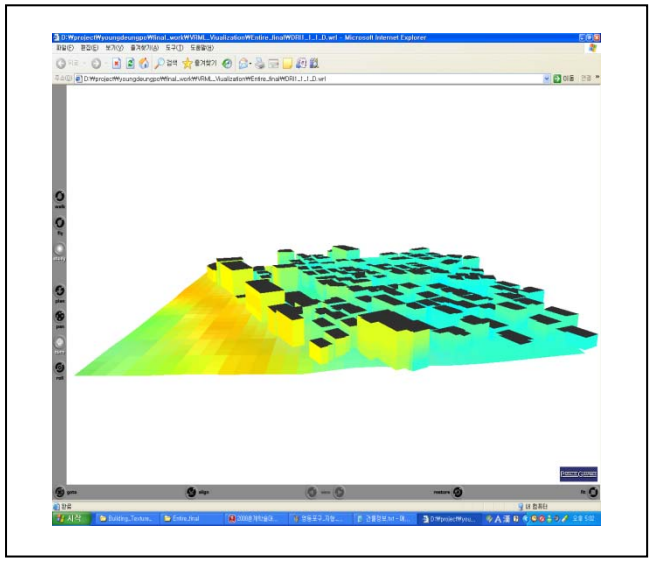


Figure 9. A snapshot of a 3D noise map

V. PERFORMANCE EVALUATION

Here, we show the performance of our approach and compare it to the performance of the approach with MPI to generate 3D noise map. The reason why we do this comparison is that we want to be sure of the advantage and disadvantage of our approach. We have been seeking the currently best cloud computing solution to process the large amount of 3D GIS data for ubiquitous city applications. To find out the answer, we did this performance comparison.

For the performance comparison, we used a ten nodes cluster, where 8 nodes had Dual Core Intel processor and 2 nodes had Quad Core Intel Processor and each node had 4 GB memory. Each node of the cluster was connected through a giga-bit Ethernet switch, runs a Ubuntu Linux 9.04 Server edition and used our own private Cloud based on OpenNebula. The JVM version 1.6.0_20 was used for Hadoop and the gcc version 4.4.1 compiler and MPICH2 were used for the MPI. For the noise map area, we selected Yeongdeungpogu District, Seoul, Korea, as shown in Figure 5, where the area size is about 24.5km^2 and the volume of the processed data was 250 GB. We processed both MapReduce and MPI experiments and measured the performance. We ran them 10 times and averaged the results. Figure 10 shows the performance and we know that the MPI case is faster than the MapReduce case.

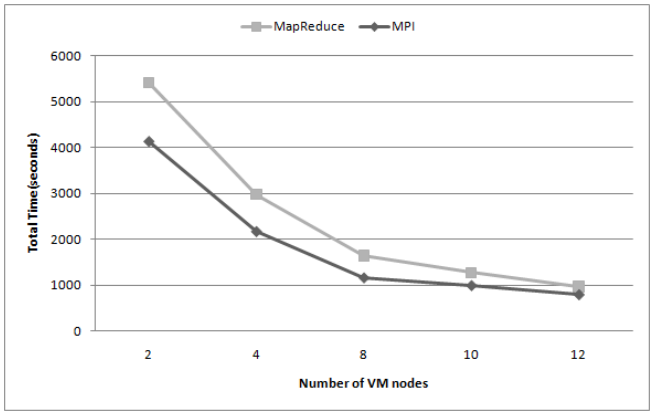


Figure 10. Performance comparison between MapReduce and MPI

Distributed and parallel processing based on message passing infrastructures such as PVM [11] and MPI [12] supports fine-grained parallelism, while workflow frameworks such as Kepler [13] and Taverna [14] supports coarse-grained parallelism. MapReduce also supports fine grained parallelism but it is different from MPI or PVM since it does not support any shared files but supports local files only. That is, by restricting the programming model, the MapReduce framework enables us to partition the given tasks into a large number of fine-grained sub-tasks, but it does not communicate each node since it only supports local files.

While MPI supports a wide variety of communication topologies for various kinds of distributed and parallel models. MapReduce only allows a communication topology from map to reduce. However, MapReduce allows us to use

 ISBN: 978-1-61208-106-9

a simple but convenient cloud computing environment, which eventually allows us to implement parallelism to run our applications. Also, MapReduce gives better support to quality of services such as fault tolerance and monitoring in data intensive parallel applications.

VI. CONCLUSION

In this paper, we have presented our cloud computing approach to process a large amount of 3D GIS data to make the 3D noise map. We find that MapReduce with Hadoop is useful to reduce the turnaround time vastly. We also present the performance comparison when we use MPI instead of MapReduce and Hadoop. We find that the MPI case is faster than the case of MapReduce with Hadoop. However, we also find that MapReduce case has better fault-tolerance and more stable than MPI case in our experiment. We find that the MapReduce with Hadoop is not suitable for real-time interactive processing and thus have been studying real-time interactive processing of our work with MapReduce and any other useful cloud computing technology.

ACKNOWLEDGMENT

This study was supported by the Seoul Research and Business Development Program (10561), Smart (Ubiquitous) City Consortium and Seoul Grid Center. We would like to give thanks to Mr. Cheol Sang Yoon, Mr, Seung Woo Rho, Mr. Chang Won LEE, Mr. Kyoung Kyu LEE, Mr. Eui Dong Hwang, Mr. Sung Min Kim and the staffs of Seoul Grid Center and the members of Smart (Ubiquitous) City Consortium for their contribution to this research.

REFERENCES

[1] DIRECTIVE 2002/49/EC OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 25 June 2002 relating to the assessment and management of environmental noise, “Official Journal of the European Communities”, 2002.

[2] Kurakula, V., “A GIS-Based Approach for 3D Noise Modelling Using 3D City Models”, MSc proposal, University of Southampton, UK, 2007.

[3] S. Oh, I. LEE, S. Tanathong, J. Ko, S. Chang, and T. Kim, “Generation of 3D Noise Map using a City Model”, 3D Geoinfo, pp. 155-160, 2008.

[4] J. Dean and S. Ghemaway, “MapReduce: Simplified Data processing on Large Clusters”, Communications of the ACM, vol. 51, January, 2008, pp. 107-113, doi:10.1145/1327452.1327492.

[5] Apache Hadoop Homepage [online], October 2010, Available from: http://hadoop.apache.org/common/.

[6] The noise map in Europe [online], October 2010, Available from: http://www.xs4all.nl/~rigolett/ENGELS/maps/.

[7] Cho, D. S., J. H. Kim, and D. Manvell. “Noise mapping using measured noise and GPS data”. Applied acoustics, vol.68 no.9, pp. 1054-1061, 2007, doi:10.1016/j.apacoust.2006.04.015.

[8] Oh, S., I. LEE, S. Kim, and K. Choi. “Generation of a Spatial city model using a Digital Map and Draft Maps for a 3D Noise Map”. Korean journal of remote sensing, vol.24 no.2, 2008, pp. 3-14.

[9] N. Golpayegani and M. Halem, “Cloud Computing for Satellite Data Processing on High End Compute Clusters”, Proceedings of the 2009 IEEE International Conference on Cloud Computing, 2009, pp. 88-92, doi: 10.1109/CLOUD.2009.71.

[10] F. Darema, “The SPMD model: past, present and future” Recent Advances in Parallel Virtual Machine and Message Passing Interface, 8th European PVM/MPI Users' Group Meeting, Santorini/Thera, Greece, 2001, Proceedings. Lecture Notes in Computer Science, 2001, Volume 2131/2001, pp. 1, doi: 10.1007/3-540-45417-9_1.

[11] Jack J. Dongarra, G. A. Geist, Robert Manchek, and V. S. Sunderam, “Integrated PVM Framework Supports Heterogeneous Network Computing” Computers in Physics, 1993, pp. 166-175.

[12] MPI (Message Passing Interface) [online], October 2010, Avaulable from: http://www-unix.mcs.anl.gov/mpi/.

[13] Bertram Ludäscher, Ilkay Altintas, Chad Berkley, Dan Higgins, Efrat Jaeger, Matthew Jones, Edward A. Lee, Jing Tao, and Yang Zhao, Scientific workflow management and the Kepler system: Research Articles. Concurr. Comput. : Pract. Exper. 18(10), pp. 1039-1065, 2006.

[14] Hull, D., K. Wolstencroft, et al. “Taverna: a tool for building and running workflows of services” Nucleic Acids Res 34(Web Server issue): W729-32, 2006.

 ISBN: 978-1-61208-106-9

Middleware for a CLEVER Use of Virtual Resources in Federated Clouds

Francesco Tusa, Maurizio Paone, Massimo Villari and Antonio Puliafito
Università degli Studi di Messina, Facoltà di Ingegneria
Contrada di Dio, S. Agata, 98166 Messina, Italy.
e-mail: {ftusa,mpaone,mvillari,apuliafito}@unime.it

***Abstract*—Nowadays Cloud Computing is becoming an interesting distributed computation infrastructure able to strongly leverage the concept of Virtualization of physical resources. This paper deals with the opportunity for managing Virtualization Infrastructures in Federated scenarios. In particular, the middleware we are introducing presents several features enabling an useful and easy management of private/hybrid clouds and provides simple and easily accessible interfaces to interact with different "interconnected" clouds. In that context, one of the main challenges it is necessary to address is the capability that systems need to interact together, maintaining separated their own domains and the own administration policies. A Cloud middleware has been designed, we named it CLEVER, and this paper describes the architecture in each part. Several UML schemas highlight the relevant complexity of our new architecture. In the current status of the work, a primitive prototype, integrating some features of the whole architecture, has been developed as far software classes implementing the basic functionalities.**

***Keywords*-Cross Cloud Computing; XMPP; Fault Tolerance; Virtual Infrastructure Management; clusters.**

I. INTRODUCTION

Cloud computing is generally considered as one of the more challenging topic in the Information Technology (IT) world, although it is not always fully clear what its potentialities are and which are all the involved implications. Many definitions of cloud computing are presented and many scenarios exist in literature. In [1], Ian Foster describes Cloud Computing as a large-scale distributed computing paradigm that is driven by economies of scale, in which a pool of abstracted, virtualized, dynamically-scalable, managed computing power, storage, platforms, and services are delivered on demand to external customers over the Internet. Until now, such trend has brought the steady rising of hundreds of independent, heterogeneous cloud providers managed by private subjects yielding various services to their clients.

In order to provide a flexible use of resources, cloud computing delegates its functionalities in a virtual context, allowing to treat traditional hardware resources like a pool of virtual ones. In addition virtualization enables the ability of migrating resources, regardless of the underlying real physical infrastructure. Peter Mell and Tim Grance from the Computer Security Division of the National Institute of Standards and Technology (NIST) are initiating important government efforts to shape essential components in the broader cloud arena[2]. They identified clouds provide services at three different levels: Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS):

- *SaaS*: Software as a Service represents the capability given to the consumer in using provider's applications running on a cloud infrastructure and accessible from various client devices through a thin client interface such as a Web browser.
- *PaaS*: Platform as a Service represents the capability given to the consumer in order to deploy his own application onto the cloud infrastructure using programming languages and tools supported by the provider (i.e Java, Python, .Net).
- *IaaS* represents the capability given to the consumer for renting processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure.

Our work is aimed to define new functionalities that focus on the lower level of services that is *IaaS*. Service Provides are customers of such infrastructures (SPs, i.e., Ebay, Facebook, Twitter, etc.); they need to easily deploy and execute their services. Owners of these infrastructures are commonly named Infrastructure Providers (IPs) or Cloud Providers (i.e. public clouds: Amazon EC2, Google, Saleforce, Microsoft Azure, RightScale, etc.). We consider the cloud as constellations of hundreds of independent, heterogeneous, private/hybrid clouds. Currently many business operators have predicted that the process toward an interoperable federated cloud scenario will begin shortly. In [3], the evolution of the cloud computing market is hypothesized in three subsequent phases:

- *Monolithic* (now), cloud services are based on proprietary architectures - islands of cloud services delivered by megaproviders (this is what Amazon EC2, Google, Salesforce and Microsoft Azure look like today);
- *Vertical Supply Chain* , over time, some cloud providers will leverage cloud services from other providers. The clouds will be proprietary islands yet, but the ecosystem building will start;
- *Horizontal Federation*, smaller, medium, and large

 ISBN: 978-1-61208-106-9

providers will federate horizontally themselves to gain: economies of scale, an efficient use of their assets, and an enlargement of their capabilities.

This paper aims to describe our architecture able to make up an interoperable heterogeneous cloud middleware that accomplishes the *Horizontal Federation*. The main challenge of interoperable clouds that needs to overcome is the opportunity that federated heterogeneous systems have to interact together, maintaining separated their own domains and administration policies. The architecture we designed is characterized by a main skeleton, in which it is possible to add more functionalities, using a flexible approach plug-in based. The diagrams we present in the next sections show several details of our design choices. The complexity of the system is rather high, this is due to the number of constrains and issues we are trying to face. The main motivation to design a new cloud middleware has been given from the lack in literature of such a middleware. The overall motivations to propose CLEVER (CLoud-Enabled Virtual EnviRonment) have been provided in our recent work [4]. That work also provides an useful and detailed description of pros and cons in CLEVER against the other existing infrastructures.

The paper is organized as follows. In Section II, we provide a brief description about the new Cloud stack. In the same section we briefly explore the current state-of-the-art in Cloud Computing and existing middleware implementations. This section critically analyses the features of such cloud middlewares and motivates the need to design and implement a new one. Section III introduces CLEVER as the *Virtual Infrastructure Manager* and explains the functional and non functional requirements that it tries to meet. Section IV provides an overview of the CLEVER's features, which are then deeply discussed in Section V where also a logical description of each middleware module is reported together with a brief description of our prototype implementation (see Sec. VI). Section VII concludes the paper.

II. Background and Related Works

The work we are describing deals with the technology necessary to make up a cloud infrastructure whatever level it is: *IaaS, PaaS and SaaS*. This technology is the well-known as *Virtualization*. Cloud Computing strongly exploits the concept of virtualization of physical resources (hosted in IaaS), through the Instantiation of Virtual Machines (VMs). During our dissertation we named these VMs as Virtual Environments (VEs), a more general term useful to describe other virtual containers (i.e., Java Virtual Containers). These VEs can be seen as the smaller part of cloud systems: "atoms", in which IT Services (i.e., PaaS or SaaS) are confined into. CLEVER is a middleware able to manage VEs in cross cloud environments. This middleware should accomplish an important core of a general framework able to address Federation among sites, guaranteeing Fault Tolerance, Easy Configuration (Zero-conf), Accounting and Billing, Performance (SLAs), Security, etc.

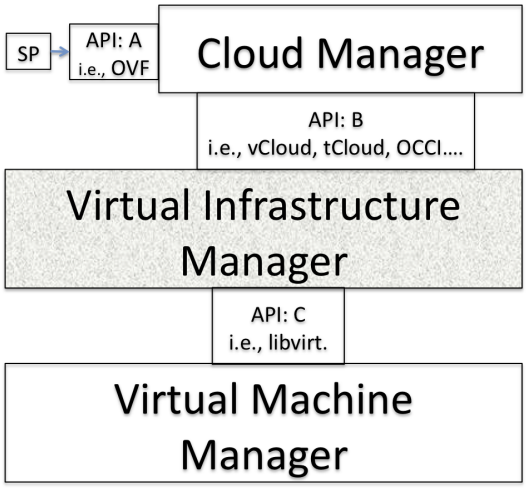


Figure 1. Cloud Management Stack

In order to identify the main components constituting a cloud and better explain the federation idea on which our work is based, we are considering the internal architecture of each cloud as the three-layered stack [5] presented schematically in Fig. 1. Starting from the bottom, we can identify: *Virtual Machine Manager, (VMM)*, *Virtual Infrastructure (VI) Manager* and *Cloud Manager, (CM)*. The Virtual Machine Manager is the layer in which the hardware virtualization is accomplished. It hides the physical characteristics of a computing platform from SPs, instead showing another abstract computing platform. The software that controls the virtualization used to be called a "control program" at its origins, but nowadays the term *hypervisor* is preferred. The main hypervisors currently used to virtualize hardware resources are: Xen [6], KVM [7], VMware [8], VirtualBox [9], Microsoft Hyper-V [10], Oracle VM [11], IBM POWER Hypervisor (PR/SM) [12], Apple Parallel Server [13], etc. For example one type of a VMM can be a physical machine with the Xen hypervisor para-virtualizer [14] controlling it (in this case the VM are Xen domains), whereas another type can be a machine with the necessary software to host KVM (full-virtualizer [15]), and so on. The VI manager is a fundamental component of private/hybrid clouds acting as a dynamic orchestrator of VEs, which automates VEs setup, deployment and management, regardless of the underlying level. The Cloud Manager layer is instead able to transform the existing infrastructure into a cloud, providing cloud-like interfaces and higher-level functionalities for security, contextualization and VM disk image management.

Recently, many Virtual Infrastructure Managers are appearing in literature, some of them can be listed as follows: OpenNebula [16], OpenQRM [17], Nimbus [18], Eucalyptus [19], etc.

 ISBN: 978-1-61208-106-9

The architectures listed above were conceived considering two main philosophies:

- Apply the virtualization model to simplify the management of proprietary datacenters.
- Reorganize the previous distributed computation middleware for accomplishing an easy management of virtualization capabilities.

The early case represents a simpler way to organizes hardware resources hosted in local and private datacenters that need a tedious and complex management (i.e., OpenQRM and Eucalyptus). In the latter, we have infrastructures in which the original focus is modified to be adapted for clouds; for instance rearranging of GRID infrastructure as well as Open Nebula and Nimbus. For instance OpenNEbula (ONE) can be considered as evolution of a Grid Manager aimed to Virtualization Systems. Thanks to the European Project RESERVOIR [20], ONE is including some key concepts we are trying to address in our work. Nimbus was originated by an adaption of Globus (GRID), in fact it is currently deployed into a Globus 4.0.x Java container.

Nowadays it is time for new cloud architectures able to deal with new cloud requirements and constrains. Furthermore these architectures must to be conceived from the scratch, that is the core of the system has to include at the beginning many more basic capabilities necessary for Cloud Computing environments. CLEVER was designed keeping mind these goals, adding as much as possible and at the early stage many functionalities. In our system, Cloud Manager layers have to offer the following opportunities: Cloud Federation, Composability, and Orchestration of resources, distributed virtual resources management, inter-cloud security, inter-cloud business model, inter-cloud networking, inter-cloud monitoring, etc. Even at Virtual Infrastructure Manager layers it is necessary to add these following features in order to satisfy the CM layers needs: local virtual resources management, load-balancing, fault tolerance, intra-cloud security, intra-cloud business models, intra-cloud networking, intra-cloud monitoring, etc. To drawing new cloud systems, it is important to initially take into account concepts of ”inter” and ”intra” requirements and capabilities. In the interoperability arena, Application Programming Interface (APIs) cover a strong role, as shown by Fig. 1 (rectangle shapes in between layers: A, B and C). APIs have to guarantee the interoperability among Clouds (see section II-B) in Private, Public and Hybrid scenarios.

Business and trading may represent the engine of Cloud development under many perspectives, Section II-A highlights these concepts.

In the direction of an open [21] and advanced cloud framework, NASA is working on an ambitious project: OpenStack [22]. On their web portal, they stated: *The goal of OpenStack is to allow any organization to create and offer cloud computing capabilities using open source software running on standard hardware. OpenStack Compute is software for automatically creating and managing large groups of virtual private servers. OpenStack Storage is software for creating redundant, scalable object storage using clusters of commodity servers to store terabytes or even petabytes of data*. These sentences remark as our overall assumptions and motivations, which led us to develop CLEVER, are completely in line with the new Cloud researching context.

In our final considerations, the figured world can appear rather complex and hard to face but the virtualization environments might greatly simplify the necessary effort. Specifically all hypervisors leverage the new virtualization capabilities offered by hardware architectures (i.e., Intel, AMD, etc.), thus allowing to limit their diversity. These features are strongly exploited by cloud operators because they increase the system performances. But at the same time thus determines the reduction of the differentiation among hypervisors, hence we can foresee a great challenge in the Virtual Computing (or Virtual Cloud). We can remark the concept considering the *virtualization world* as the simplest way to identify and regroup a subset of functionalities that it is necessary to deal within a Cloud. Since the formalization of such requirements, constrains, and capabilities might not be particularly tough to accomplish.

A. Commercial Cloud Infrastructures and their businesses.

Cloud Computing is finding a wide consensus among IT operators, because behind its computation model it is possible to advise a real business model [23]. One of the main reason of the Grid Computing failure, seen under the economic perspectives, was the not applicability of any business model on it. Currently in the IT world we are assisting to a progressive mixing of meaningful efforts among operators, scientific communities and organizations for standards (as well: DTMF, IEEE, IETF, ITU, etc.) to enforce this new distributed computation infrastructure: Cloud Computing. We can state that it is not possible to draw clear boundaries among commercial, academic, opensource and legacy cloud platforms. Several platforms born in academia have migrated to commercial purposes (Nimbus, Eucaliptus, OpeNebula), others from legacy to OpenSource (OpenQRM) to consolidate and advertise the work done, and finally a few of them were totally born as OpenSource (OpenStack) or commercial (Citrix, VMware, Telefonica, IBM, HP, etc.) approaches. The evolution we are noting is a movement of lower layers toward the upper layers of the Cloud stack. For instance hypervisor as Vmware and Xen is providing a new cloud middleware able to address all the issues in entire distributed datacenters (VMM to VIM: VMware vSphere, XEN Cloud Platform). VIMs are looking at Cloud Management.

In this process, we think the commercial IT world might be far to define an interoperable framework, whereas the current VIM middleware should change the original focus to be ready for the new challenges. A platform ready to

 ISBN: 978-1-61208-106-9

face these challenges might be the OpenStack framework, it promises an interesting contribution, but it is a young architecture yet. The CLEVER infrastructure might also provide a valid support in that direction.

The standardization rate of proposals is also showing an huge interest of IT stockholders, the next section will provide a brief enlightenment on that.

B. Cloud Standards in a Nutshel: APIs

The concepts of interoperability need to be enforced at Cloud Manager level, but the VIMs need to provide Application Program Interfaces (APIs) enabling the full control and monitoring of cross-cloud virtual resources. The federation among Clouds can be perpetrated if each Cloud Operator can dynamically join the federation [24], interact with federated clouds in trustiness [25],[26] and finally exchange data with their partners [27]. Since the federation can exist if there is a high level infrastructure that allows this aggregation. Furthermore it is not possible to spread computation among several IPs, without an adequate intercommunication protocol among parties. Fig. 1 highlights this concept. In the Cloud stack it is necessary to include APIs among each layer that allow to cross-correlate more heterogeneous infrastructures. The working group in DMTF Standards is defining the Cloud Incubator Initiative for Cloud Management Standards. It was formed to address management interoperability for Cloud Systems [28]. The DTMF organization has began the initiatives in Virtualization environments with Open Virtual Format (OVF, see the fig., API: A).The format (OVF) represents an early descriptor able to define the customer requirements, in terms of number of VEs to instantiate; memory, CPUs, storage and etc. to allocate and so on. This format permits the interaction from IaaSs and their costumers (SPs).

Inside DTMF many IT companies are also trying to add standards in the cloud particularly useful for their businesses. For instance VMware has introduced in DMTF the VCloud[29] API, while Telefonica their TCloud [30] API. Both proposals should guarantee a set of new parameters useful for the cross interaction. In this way, many more functionalities should be addressed, such as: load balancing, fault tolerance (VMs replication), network configuration, firewalling policies, etc. The main feature of such standards is the adoption of OVF as the base, to meet the SPs requirements and constrains, without an heavy translation among different descriptor files. Another example of API standardization process is the Open Cloud Computing Interface (OCCI) standard [31]. The OCCI standard, is proposed inside the Open Grid Forum (OGF), it should guarantee an exposition of VIM capabilities, as depicted in Fig. 1 (API: B). But it does not appear to be nor particularly flexible neither oriented to SPs requirements. In this case it needs a translation from OVF. All the standards presented above are based on XML in order to guarantee the portability and interoperability in heterogeneous systems.

In the following subsection we briefly describe the main VIMs. Subsequently, we provide an in-depth description of our CLEVER architecture.

C. Related Works

As we introduced in Section II and stated in [5], cloud management can be performed at the lowest stack layer of Fig. 1 as *Virtual Infrastructure Management*.

The project OpenQRM [17] is an open-source platform for enabling flexible management of computing infrastructures. It is able to implement a cloud with several features that allows the automatic deployment of services. It supports different virtualization technologies and format conversion during migration. This means VEs (appliances in the OpenQRM terminology) can not only easily move from physical to virtual (and back), but they can also be migrated from different virtualization technologies, even transforming the server image. OpenQRM is able to grant a complete monitor of systems and services by means of the Nagios tool [32], which maps the entire openQRM network and creates (or updates) its corresponding configuration (i.e., all systems and available services). Finally, OpenQRM addresses the concepts related to High Availability (HA) systems: virtualization is exploited to allow users to achieve services failover without wasting all the computing resources (e.g. using stand-by systems).

OpenNebula [16] is an open and flexible tool to build a Cloud computing environment. OpenNebula can be primarily used as a virtualization tool to manage virtual infrastructures in a data-center or cluster, which is usually referred as Private Cloud. Only the more recent versions of OpenNebula are trying to supports Hybrid Cloud to combine local infrastructure with public cloud-based infrastructure, enabling highly scalable hosting environments. OpenNebula also supports Public Clouds by providing Cloud interfaces to expose its functionalities for virtual machine, storage and network management.

Still looking at the stack of Fig. 1, other middlewares work at an higher level than the VI Manager (High-level Management) and albeit they provide high-level features (external interfaces, security and contextualization) their VI management capabilities are limited and lack VI management features: this type of cloud middlewares include Globus Nimbus [18] and Eucalyptus [19].

Nimbus [18] is an open source toolkit that allows to turn a set of computing resources into an Iaas cloud. Nimbus comes with a component called workspace-control, installed on each node, used to start, stop and pause VMs, implements VM image reconstruction and management, securely connects the VMs to the network, and delivers contextualization. Nimbus's workspace-control tools work with Xen and KVM but only the Xen version is distributed. Nimbus provides

 ISBN: 978-1-61208-106-9

interfaces to VM management functions based on the WSRF set of protocols. There is also an alternative implementation exploiting Amazon EC2 WSDL.

Eucalyptus [19] is an open-source cloud-computing framework that uses the computational and storage infrastructures commonly available at academic research groups to provide a platform that is modular and open to experimental instrumentation and study. Eucalyptus addresses several crucial cloud computing questions, including VM instance scheduling, cloud computing administrative interfaces, construction of virtual networks, definition and execution of service level agreements (cloud/user and cloud/cloud), and cloud computing user interfaces.

III. THE CLEVER ARCHITECTURE

CLEVER aims to provide *Virtual Infrastructure Management* services and suitable interfaces at the *High-level Management* layer to enable the integration of high-level features such as Public Cloud Interfaces, Contextualization, Security and Dynamic Resources provisioning.

Looking at the middleware implementations, which act as *High-level Cloud Manager* [18], [19], it can be said that their architecture lacks modularity: it could be a difficult task to change these cloud middleware for integrating new features or modifying the existing ones. CLEVER instead intends granting an higher scalability, modularity and flexibility exploiting the plug-ins concept. This means that other features can be easily added to the middleware just introducing new plug-ins or modules within its architecture without upsetting the organization.

Furthermore, analysing the current existing middleware [17], [16], which deal with the *Virtual Infrastructure Management*, we retain that some new features could be added within their implementation in order to achieve a system able to grant high modularity, scalability and fault tolerance. Our idea of cloud middleware, in fact, finds in the terms flexibility and scalability its key-concepts, leading to an architecture designed to satisfy the following requirements: 1) *persistent communication* among middleware entities; 2) *transparency* respect to "user" requests; 3) *fault tolerance* against crashes of both physical hosts and single software modules; 4) heavy *modular design* (e.g., monitoring operations, managing of hypervisor and managing of VEs images will be performed by specific plug-ins, according to different OS, different hypervisor technologies, etc.); 5) *scalability* and *simplicity* when new resources have to be added, organized in new hosts (within the same cluster) or in new clusters (within the same cloud); 6) automatic and optimal *system workload balancing* by means of dynamic VEs allocation and live VEs migration.

Looking at Figure 2, we believe the existing solutions lack a cloud Virtual Infrastructure able to implement all the characteristics of each row. The big black dot in the cell specifies the feature that the middleware has. CLEVER has

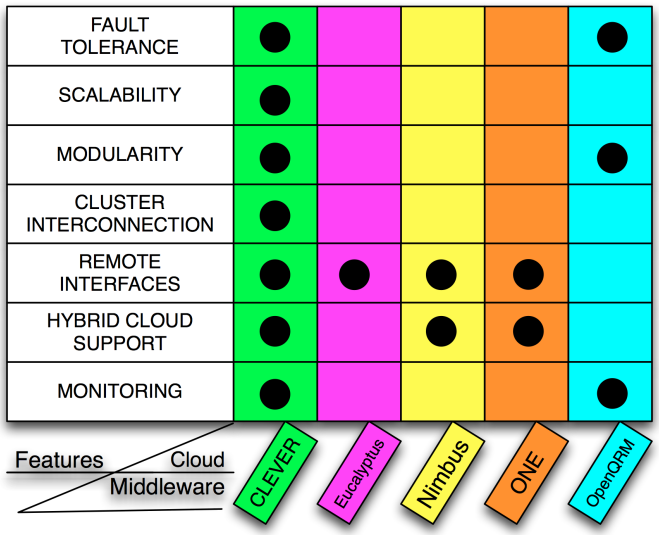


Figure 2. A comparison of CLEVER features VS other Cloud middleware implementations

all the features listed, in fact CLEVER is able to manage in a flexible way both physical infrastructures composed of several hosts within a cluster and physical infrastructures composed of different "interconnected" clusters. This task is performed ensuring fault tolerance while operations are executed, exploiting particular methods which allow the dynamic activation of recovery mechanisms when a crash occurs. Furthermore, due to its pluggable architecture, CLEVER is able to provide simple and accessible interfaces that could be used to implement the concept of hybrid cloud. Finally, it is also ready to interact with other different cloud technologies supposing that their communication protocol or interfaces are known.

IV. CLEVER REFERENCE SCENARIO

Our reference scenario consists of a set of physical hardware resources (i.e., a cluster) where VEs are dynamically created and executed on the hosts considering their workload, data location and several other parameters. The basic operations our middleware should perform refer to: 1) Monitoring the VEs behavior and performance, in terms of CPU, memory and storage usage; 2) Managing the VEs, providing functions to destroy, shut-down, migrate and network setting; 3) Managing the VEs images, i.e., images discovery, file transfer and uploading.

Considering the concepts stated in [4] and looking at Fig. 1, such features, usually implemented in the *Virtual Infrastructure Management* layer, can be further analyzed and arranged on two different sub-layers: *Host Management* (lower) and *Cluster Management* (higher).

Grounding the design of the middleware on such logical subdivision and taking into account the satisfaction of all

 ISBN: 978-1-61208-106-9

the above mentioned requirements, the simplest approach to design our middleware is based on the architecture schema depicted in Fig. 3, which shows a cluster of n nodes (also an interconnection of clusters could be analyzed) each containing a *host level* management module (Host Manager). A single node may also include a *cluster level* management module (Cluster Manager). All these entities interact exchanging information by means of the *Communication System* based on the XMPP. The set of data necessary to enable the middleware functioning is stored within a specific *Database* deployed in a distributed fashion.

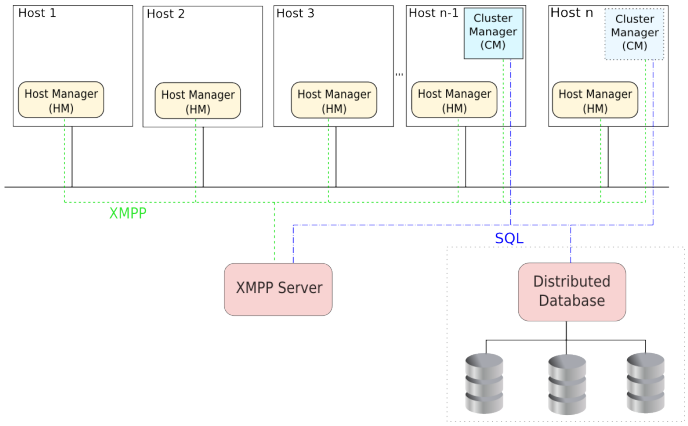


Figure 3. CLEVER reference Scenario representation

Figure 3 shows the main components of the CLEVER architecture, which can be split into two logical categories: software agents (typical of the architecture itself) and the tools they exploit. To the former set belong both *Host Manager* and *Cluster Manager*:

- **Host manager (HM)** performs the operations needed to monitor the physical resources and the instantiated VEs; moreover, it runs the VEs on the physical hosts (downloading the VE image) and performs the migration of VEs (more precisely, it performs the low level aspects of this operation). To carry out these functions it must communicate with the hypervisor, hosts' OS and distributed file-system on which the VE images are stored. This interaction must be performed using a plug-ins paradigm.
- **Cluster Manager (CM)** acts as an interface between the clients (software entities, which can exploit the cloud) and the HM agents. CM receives commands from the clients, performs operations on the HM agents (or on the database) and finally sends information to the clients. It also performs the management of VE images (uploading, discover, etc.) and the monitoring of the overall state of the cluster (resource usage, VEs state, etc.). Following our idea, at least one CM has to be deployed on each cluster but, in order to ensure higher fault tolerance, many of them should exist. A master CM will exist in active state while the other ones will remain in a monitoring state, although client messages are listened whatever operation is performed.

Regarding the tools such middleware components exploit, we can identify the *Distributed Database* and the *XMPP Server*:

- **Distributed Database** is merely the database containing the overall set of information related to the middleware (e.g. the current state of the VEs or data related to the connection existing on the Communication System). Since the database could represent a centralized point of failure, it has to be developed according to a well structured approach, for enabling fault tolerance features. The best way to achieve such features consists of using a Distributed Database
- **XMPP Server** is the "channel" used to enable the interaction among the middleware components. In order to grant the satisfaction of our requirements, it is able to offer: decentralization (i.e., no central master server should exist: such capability in native on the XMPP) in a way similar to a p2p communication system for granting fault-tolerance and scalability when new hosts are added in the infrastructure; flexibility to maintain system interoperability; security based on the use of channel encryption. Since the XMPP Server also could exploit the distributed database to work, the solution enables an high fault tolerance level and allows system status recovery if a crash occurs.

V. ARCHITECTURE DESIGN

In this Section, we will provide further details about the internal structure of the two main software components deployed on the cluster's hosts (already pointed out in Fig. 3) analyzing the Host Manager and the Cluster Manager. Moreover, the main results of our design process will be presented using the UML description.

Figure 4 shows the internal organization of such CLEVER components and how they are deployed on the reference scenario already introduced.

The left part of the Figure points out the Host Manager. As previously stated, such component lies in the lower part of the VI layer of the stack and interacts with the OS of each host of the cluster. The main modules composing the Host Manager are described below.

- **Monitor**: Provides resource usage monitoring for each host. The information are organized and made available to the HM coordinator.
- **Hypervisor Interface**: is the middleware back-end to the hypervisor running on the host's OS. Different virtualization technologies could be employed using different plug-ins structure has to be developed.
- **Image Manager**: supplies to the Hypervisor Interfaces the needed VE disk-image corresponding to a specific

 ISBN: 978-1-61208-106-9

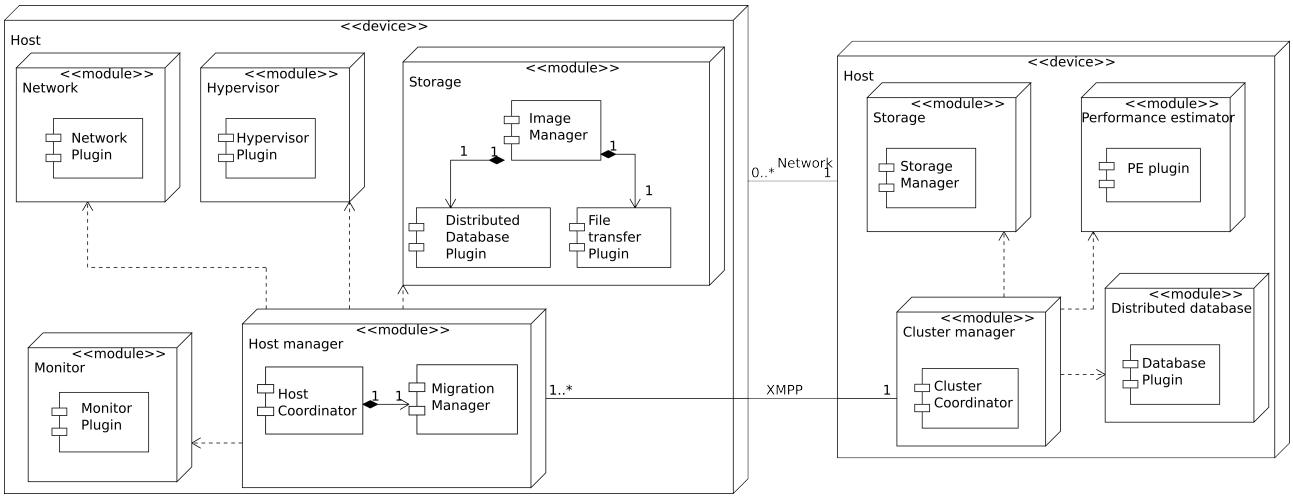


Figure 4. CLEVER deployment diagram

VE. Different plug-ins associated could be associated to different data access/transfer methods.

- **Network Manager**: Gathers information about the host network state. Manages host network (OS level) according to the guidelines provided by the HM Coordinator: dynamically creates network bridges, routing and firewalling rules.

The Cluster Manager, depicted in the right part of Figure 4, lies in the higher part of the same stack layer of the Host Manager and coordinates all the entities of the middleware (i.e. the HMs). Its internal composition is described in the following.

- **Database Manager**: interacts with the database employed to store information needed to the cluster handling. Database Manager must maintain the data strictly related to the cluster state.
- **Performance Estimator**: Analysis of the set of data collected from the Coordinator, in order to compute and provide a probable trend estimation of the collected measures.
- **Image Manager**: manages both registrations and uploads within the Cluster Storage System of the VEs disk-images. The Storage Manager is used to perform the registration process of such files and manage the internal cluster distributed file system.

As the Figure points out, both Host Manager and Cluster Manager include a specific module named respectively Host Coordinator and Cluster Coordinator. More specifically, the Host Coordinator manages the communication between the Host Manager internal modules while the Cluster Coordinator performs the same task for the Cluster Manager modules.

Furthermore, the Host Coordinator and Cluster Coordinator, exploiting the XMPP connection, allows the middleware functioning by exchanging messages in a chat-like fashion: as previously introduced, both Host Manager and Cluster Manager(s) will attend a XMPP chat session for enabling operation of resource monitoring, VEs allocation. The middleware back-end to the XMPP is represented by the Host Coordinator and the Cluster Coordinator.

As will be better explained in the following, within each component of the middleware, each module has been designed according to a well-structured plugin fashion: this allows a given module to be independent from the underlying technologies of the hosts on which it is running, since the best plugin will be employed and linked (dynamically) to the corresponding module.

Figure 5, considering the reference scenario described above, presents the whole use case diagram of the middleware, highlighting the main offered services and involved actors. The latter include Host Coordinator, Cluster Coordinator, Operating System, Hypervisor, Client and (eventually) Other Clusters. Starting from such diagram which describes the middleware specification, the design process has been refined using both activity diagrams, sequence diagrams and other use case diagrams. The final result of such work has been the definition of the whole system regarding the classes (referring to the object oriented software development) of each module on the CLEVER components.

VI. Prototype Implementation

In the current status of the work, a primitive prototype, integrating some features of the whole architecture, has been developed as far software classes implementing the basic functionalities of the Host Manager and Cluster Manager have been written using the Java programming language, allowing middleware components interaction by means of the XMPP protocol: the management of all the problems related to identification, access, monitoring and control of hardware resources in the Cloud Environment have been thus addressed in the current implementation.

 ISBN: 978-1-61208-106-9

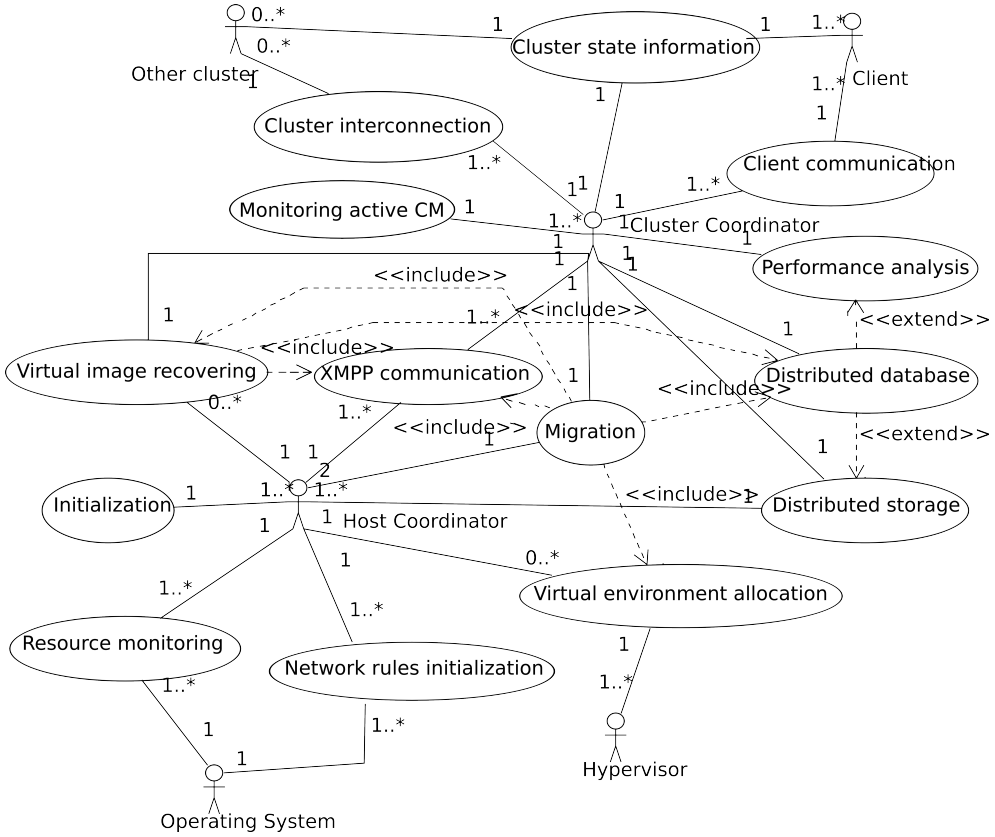


Figure 5. CLEVER general use case diagram

As previously introduced, using the UML description we achieved a deep description of the middleware behavior. The design process has lead to the definition of a series of class, packaged according to the deployment diagram of Fig. 4, each referred to a different module of the middleware components: currently, our set class consists of approximately 150 hundreds of classes and 50% of them have been fully implemented.

Figure 6 depicts the class diagram of the Hypervisor Interface module: we would like to underline that in our current implementation, each module of both the HM and the CM has been developed as a self-contained entity running into a different OS process. Such approach ensures an higher fault-tolerance level (since a failure of a single process/module will be isolated) and increases the modularity of the whole system. A drawback of such approach is related to the high level of complexity introduced: since each module within the middleware components represents a different process, a specific (interprocess) communication method has to be employed to ensure the interaction between all the modules. According to the aforementioned plugin approach, our communication method is based on a particular plugin referring to Apache ActiveMQ [33] and Java Message Service.

According to the description reported in the Section III, both the Host Manager and the Cluster Manager have been implemented including a Java class which act as XMPP client exploiting the Smack [34] set of libraries. By means of such software module, our middleware components are able to communicate each other exploiting the XMPP facilities provided by the Ejabber XMPP server [35]. The latter has been chosen due to its distributed and fault tollerance features.

In order to manage VEs allocation, our implementation of the HM components, includes a specific software module whose aim refers to the interaction with the hypervisor running on the Host OS: such software practically implements a plugin for the Hypervisor Interface of the Host Manager, linking the Libvirt [36] set of libraries. Using the API offered by libvirt, the plugin is able to start create and start virtual environment on several hypervisor.

VII. Conclusions and Future Works

In this work, we described the design principles and the preliminary prototype implementation of our cloud middleware named CLEVER: unlike similar works existing in the literature, CLEVER provides both *Virtual Infrastructure Management* services and suitable interfaces at the *High-level Management* layer to enable the integration of Public

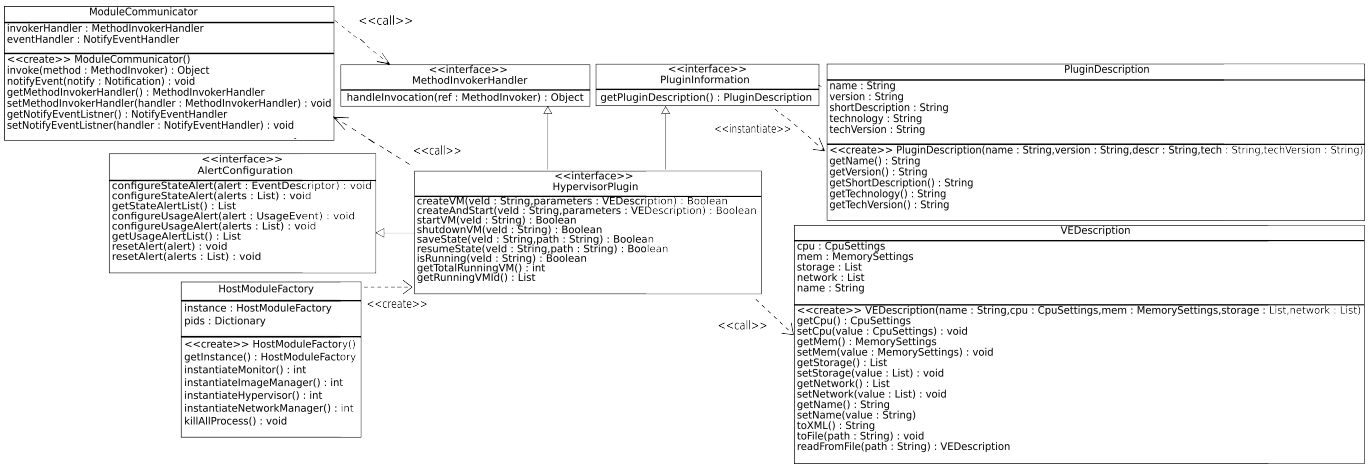


Figure 6. CLEVER: example of the hypervisor interface class diagram

Cloud Interfaces, Contextualization, Security and Dynamic Resources provisioning within the cloud infrastructure.

Furthermore, thanks to its pluggable design, CLEVER grants scalability, modularity, flexibility and fault tolerance. We are working to further extend the middleware functionalities according to the reference UML model described in this paper. Moreover, a set of tests is being executed to obtain a comprehensive set of experimental results to deeply evaluate the behavior of the middleware and its performance.

VIII. ACKNOWLEDGEMENTS.

The research leading to the results presented in this paper has received funding from the European Union's seventh framework programme (FP7 2007-2013) Project RESERVOIR under grant agreement number 215605.

REFERENCES

[1] 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*, pp. 1–10, 2008.

[2] National Institute of Science and Technology. NIST Definition of Cloud Computing; http://csrc.nist.gov/groups/SNS/cloud-computing/ July 2010.

[3] T. Bittman, "The evolution of the cloud computing market," *Gartner Blog Network,http://blogs.gartner.com/thomas_bittman/2008/11/03/the-evolution-of-the-cloud-computing-market/*, November 2008.

[4] F. Tusa, M. Paone, M. Villari, and A. Puliafito., "CLEVER: A CLoud-Enabled Virtual EnviRonment," in *15th IEEE Symposium on Computers and CommunicationsS Computing and Communications, 2010. ISCC '10. Riccione*, June 2010.

[5] B. Sotomayor, R. Montero, I. Llorente, and I. Foster, "Virtual Infrastructure Management in Private and Hybrid Clouds," *Internet Computing, IEEE*, vol. 13, pp. 14–22, Sept.-Oct. 2009.

[6] Xen Hypervisor - Leading open source hypervisor for servers. http://www.xen.org/ August 2010.

[7] KVM (for Kernel-based Virtual Machine) is a full virtualization solution for Linux on x86 hardware. http://www.linux-kvm.org August 2010.

[8] Virtualization: the essential catalyst for enabling the transition to secure cloud computing. http://www.vmware.com/it/ August 2010.

[9] x86 virtualization software package developed by Sun Microsystems. http://www.virtualbox.org/ August 2010.

[10] Microsoft Hyper-V Server 2008 R2, the stand-alone product that provides a reliable and optimized virtualization solution. http://www.microsoft.com/hyper-v-server/en/us/default.aspx August 2010.

[11] Oracle VM, ther virtualization software that fully supports both Oracle and non-Oracle applications, and delivers more efficient performance. http://www.oracle.com/us/technologies/virtualization/oraclevm/ August 2010.

[12] The POWER Hypervisor, the abstraction layer between the hardware and firmware and the operating system instances for GX host channel adapter (HCA) implementations. http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/ August 2010.

[13] The Mac: Real versatility through virtualization. http://www.apple.com/business/solutions/it/virtualization.html August 2010.

[14] Paravirtualization. The virtualization technique that presents a software interface to virtual machines that is similar but not identical to that of the underlying hardware. http://en.wikipedia.org/wiki/Paravirtualization July 2010.

[15] Full Virtualization. The virtualization technique used to provide a certain kind of virtual machine environment, that is a complete simulation of the underlying hardware. http://en.wikipedia.org/wiki/Full-virtualization July 2010.

 ISBN: 978-1-61208-106-9

[16] B. Sotomayor, R. Montero, I. Llorente, and I. Foster, "Resource Leasing and the Art of Suspending Virtual Machines," in *High Performance Computing and Communications, 2009. HPCC '09. 11th IEEE International Conference on*, pp. 59–68, June 2009.

[17] OpenQRM official site: http://www.openqrm.com July 2010.

[18] C. Hoffa, G. Mehta, T. Freeman, E. Deelman, K. Keahey, B. Berriman, and J. Good, "On the Use of Cloud Computing for Scientific Workflows," in *SWBES 2008, Indianapolis*, December 2008.

[19] 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*, pp. 124–131, May 2009.

[20] B. R. J. Caceres, R. Montero, "Reservoir: An architecture for services,," The first issue of the RESERVOIR Architecture document. http://www.reservoir-fp7.eu/twiki/pub/Reservoir/Year1 Deliverables/080531-ReservoirArchitectureSpec-1.0.PDF, June 2008.

[21] Open Cloud Manifasto, dedicated to the belief that the cloud should be open, http://www.opencloudmanifesto.org/ July 2010.

[22] The Open Source, Open Standards Cloud, Innovative, open source cloud computing software for building reliable cloud infrastructure. http://openstack.org/ July 2010.

[23] Sun Microsystems, Take your business to a Higher Level - Sun cloud computing technology scales your infrastructure to take advantage of new business opportunities, guide, April 2009.

[24] A. Celesti, M. Villari, and A. Puliafito, "Ecosystem Of Cloud Naming Systems: An Approach For The Management And Integration Of Independent Cloud Name Spaces.," in *The 9th IEEE International Symposium on Network Computing and Applications (IEEE NCA10), Boston, USA*, July 2010.

[25] A. Celesti, F. Tusa, M. Villari, and A. Puliafito., "Three-phase Cross-cloud Federation Model: The Cloud Sso Authentication.," in *The 2nd International Conference on Advances in Future Internet (AFIN 2010), Venice, Italy*, July 2010.

[26] E. Bertino, F. Paci, R. Ferrini, and N. Shang, "Privacy-preserving digital identity management for cloud computing," *Computer*, vol. 32, pp. 21–27, March 2009.

[27] A. Celesti, F. Tusa, M. Villari, and A. Puliafito, "How To Enhance Cloud Architectures To Enable Cross-federation.," in *The 3rd IEEE International Conference on Cloud Computing (IEEE Cloud 2010), Miami, Florida, USA*, July 2010.

[28] DMTF Cloud Management Standards Workgroup Formed to Address Management Interoperability for Cloud Systems. http://www.dmtf.org/about/cloud-incubator July 2010.

[29] VMware vCloud Cloud Computing For Any Application, Any Customer. http://www.vmware.com/products/vcloud/ July 2010.

[30] Telefonica I+D releases the TCloud API for cloud Computing interoperability. April 2010 http://www.tpd.com.br/en/News

[31] Open Cloud Computing Interface WG (OCCI-WG) http://www.ogf.org/ July 2010.

[32] Nagios, The Industry Standard in IT Infrastructure Monitoring: http://www.nagios.org/ July 2010.

[33] Apache ActiveMQ: Open source messaging and Integration Patterns provider, http://activemq.apache.org/ July 2010.

[34] Smack client API, the Open Source XMPP (Jabber) client library for instant messaging and presence. http://www.igniterealtime.org/projects/smack/ July 2010.

[35] Ejabberd, the Erlang Jabber/XMPP daemon, http://www.ejabberd.im/ July 2010.

[36] Libvirt API http://libvirt.org/ July 2010.

 ISBN: 978-1-61208-106-9