Jörn Altmann
Omer F. Rana (Eds.)

Economics of Grids, Clouds, Systems, and Services

LNCS 6296

7th International Workshop, GECON 2010
Ischia, Italy, August 2010
Proceedings

Springer

Lecture Notes in Computer Science 6296

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Jörn Altmann Omer F. Rana (Eds.)

Economics of Grids, Clouds, Systems, and Services

7th International Workshop, GECON 2010
Ischia, Italy, August 31, 2010
Proceedings

Springer

Volume Editors

Jörn Altmann
Seoul National University
College of Engineering
Department of Industrial Engineering
Technology Management, Economics, and Policy Program
599 Gwanak-Ro, Gwanak-Gu, 151-744 Seoul
South-Korea
E-mail: jorn.altmann@acm.org

Omer F. Rana
Cardiff University
School of Computer Science
Queen's Buildings
Newport Road, Cardiff CF24 3AA
UK
E-mail: o.f.rana@cs.cardiff.ac.uk

Library of Congress Control Number: 2010933594

CR Subject Classification (1998): C.2.4, K.4.4, H.4, H.3, H.5, J.1

LNCS Sublibrary: SL 5 – Computer Communication Networks and Telecommunications

ISSN 0302-9743
ISBN-10 3-642-15680-0 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-15680-9 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper 06/3180

Preface

The commercial exploitation of distributed computing technologies is slowly starting to become popular under the general area of cloud computing. These solutions allow selling and buying of resources (i.e., computing, network, software, and data resources) on demand. Existing solutions in this area are diverse, ranging from Infrastructure-as-a-Service (IaaS) models via Platform-as-a-Service (PaaS) to Software-as-a-Service (SaaS) models. Although the economics of these services is not yet fully understood and the interoperability between such services is still lacking, a common market for computing services is slowly developing.

Such a market would allow buyers and sellers of computing services to trade their excess capacity or make available their capacity at a cost. However, it is still not possible for a market participant to act as a resource provider or seller, or trade based on the current level of demand. Another example of a developing open market is the emergence of Web2.0-based services. These enable consumers to create new services by aggregating services from multiple providers. The benefit of these solutions is that "value" can be created by combining services at different prices.

The GECON workshop series is intended to enable researchers and practitioners from academia, industry, and national research laboratories to identify economics-related issues and solutions associated with the development of services. Such work can comprise extensions to existing technologies, successful deployments of technologies, economic analysis, and associated theoretical concepts. The purpose of this workshop is to gather original work and build a strong community in this increasingly important area of the future economy.

The 7th International Workshop on the Economics of Grids, Clouds, Systems, and Services (GECON 2010) attracted a number of high-quality paper submissions. In total, we received 19 submissions, of which 6 were accepted as full papers and another 6 as "work-in-progress" papers. Each paper was reviewed by between 3 and 5 international experts.

For the proceedings, the 12 accepted papers of this workshop have been grouped into 4 sessions – with each session consisting of 3 contributions: (1) Service Evaluation and Trust; (2) Service Pricing and Software Licenses; (3) Adoption of Grid and Cloud Services; and (4) Value Chains and Service Level Agreements. It is to be noted that there continues to be high interest in Service Level Agreements (SLAs) as important enablers for service-oriented systems, since over 40% of the papers report on the use of SLAs.

In the first session on "Service Evaluation and Trust", the contribution by Frank Dickmann et al. entitled "Technology Transfer of Dynamic IT Outsourcing Requires Security Measures in SLAs" uses a questionnaire-based approach to assess the need for security within Service Level Agreements (SLAs). The authors interviewed around 75 experts at the CeBIT fair in Germany to gather their data. It is really interesting to see a paper that discusses user perception of security and highlights the need to focus

on specific security challenges for SLAs in grid and cloud computing. The paper entitled "Service Selection Decision Support in the Internet of Services" by Konstantinos Tserpes et al. discusses how a "Quality of Experience", gained from multiple customers using a particular service, could be used to support service selection. The authors identify how collaborative filtering techniques can be used to relate user ratings, and thereby group users with similar types of ratings for services. Simulation is used to validate the approach. The final contribution in this session, entitled "Resource-Level QoS Metric for CPU-Based Guarantees in Cloud Providers" by Goiri et al. proposes a CPU allocation metric for allowing cloud resource providers to dynamically allocate their capacity for this resource among the running services depending on demand. The work is motivated by the observation that current cloud providers do not support fine-grained resource level QoS guarantees on their SLAs – with most commercial providers focusing on resource availability guarantees. The customer's CPU usage is used in the metric definition, but "fake" SLA violations are avoided when a customer's task does not use all its allocated resources.

In the second session on "Service Pricing and Software Licenses", Silagi et al. identify "A Framework for Building Intelligent SLA Negotiation Strategies under Time Constraints". The contribution makes use of an agent-based system utilizing Bayesian learning for negotiating SLA parameters under time constraints. Their work shows that setting time constraints may actually lead to better results. It forces players to learn the required parameters more quickly. A comparison with other strategies is also provided by the authors. The contribution by Rohitratana and Altmann, entitled "Agent-Based Simulation of the Software Market under Different Pricing Schemes for Software-as-a-Service and Perpetual Software" focuses on developing a simulation to support the pricing of software licenses, comparing three different schemes: derivative-follower (DF), demand-driven (DD) and competitor-oriented (CO). The simulation involves two types of agents: customer agents and vendor agents – and the authors show which of the three schemes DF, DD or CO should be followed in a particular context. The software license theme is continued in the paper by Ziegler et al. entitled "Software Licenses as Mobile Objects in Distributed Computing Environments", which focuses on supporting license management within grid computing and service-oriented environments, decoupling license usage from authorization, and expresses authorization by SLAs. The contribution focuses on supporting license management via mobile objects that do not need to be managed by a centralized server – and instead may move to the environment/host where they are needed.

The next two sessions, "Adoption of Grid and Cloud Services" and "Value Chains and Service Level Agreements", focus on work-in-progress contributions that are at an early stage of maturity. Heine and Strebel in "IaaS Adoption Determinants in Enterprises" discuss organizational challenges that have limited the uptake of Infrastructure-as-a-Service (IaaS) as an IT provisioning model. The authors use an interview-based approach – having identified 50 experts (and finally interviewing 20 of these). Oberle and Fisher in "ETSI CLOUD – Initial Standardization Requirements for Cloud Services" report on standards that are necessary for realizing future interoperable clouds. This contribution identifies the European cloud standardization landscape and the term "cloud computing", and provides a list of requirements, divided into 11 categories, about standardization issues of cloud-computing-related areas. It summarizes the out-

come of an ETSI (European Telecommunications Standards Institute) Technical Committee on Cloud Computing workshop, where experts from industry and research came together. Tobias Knoch then discusses how low resource utilization in grid computing systems could be explained by using the Inverse Tragedy of the Commons theory, in the paper "Approaching the Internalization Challenge of Grid Technologies into e-Society by e-Human Grid Ecology". In the final session, Markus Böhm et al. in their contribution "Towards a Generic Value Network for Cloud Computing" describe the transition from linear value chains to generic "value networks", identifying the role of different actors involved in a cloud computing market. The authors use an interview-based approach to identify future "value" streams within this emerging area. Petri et al. in their contribution "SLA as a Complementary Currency in Peer-2-Peer Markets" identify how SLAs can be used as a complementary currency to support resource exchange within a distributed system. They use a PeerSim-based simulation to demonstrate profit/loss that can arise within a market of collaborating peers, exchanging SLAs. Finally, Ul Haq et al. in their paper "SLA Validation in Layered Cloud Infrastructures" present an approach for combining SLAs across different infrastructures. The authors present a multimedia data sharing scenario to validate their approach.

To make this workshop a success, many people contributed to this event. In particular, we would like to express our gratitude to the organizers of the 2010 Euro-Par conference for their support in co-locating the GECON 2010 workshop at Ischia in Naples (Italy). We would also like to thank Alfred Hofmann of Springer for his help in getting the proceedings printed on time. Finally, our gratitude goes to Marcel Risch for his time and effort in setting up the website.

July 2010

Jörn Altmann
Omer Rana

Organization

GECON 2010 was organized by the Technology Management, Economics, and Policy Program, Seoul National University and the School of Computer Science, Cardiff University in collaboration with Euro-Par 2010.

Executive Committee

Chairs

Jörn Altmann	Seoul National University, South Korea
Omer F. Rana	Cardiff University, UK

Program Committee

Hermant K. Bhargava	UC Davis, USA
Ivona Brandic	Technical University of Vienna, Austria
Rajkumar Buyya	University of Melbourne, Australia
Costas Courcoubetis	Athens University of Economics and Business, Greece
Jeremy Cohen	Imperial College, UK
Dang Minh Quan	CREATE-NET, Italy
Karim Djemame	University of Leeds, UK
Torsten Eymann	University of Bayreuth, Germany
Thomas Fahringer	University of Innsbruck, Austria
Wolfgang Gentzsch	DEISA, EU
Matthias Hovestadt	Technical University of Berlin, Germany
Chun-Hsi Huang	University of Connecticut, USA
Admela Jukan	Technical University of Braunschweig, Germany
Odej Kao	Technical University of Berlin, Germany
Stefan Kirn	University of Hohenheim, Germany
Tobias A. Knoch	Erasmus University, Netherlands
Bastian Koller	HLRS, Germany
Harald Kornmayer	NEC Laboratories Europe, Germany
Ramayya Krishnan	Carnegie Mellon University, USA
Kevin Lai	HP Labs, USA
Byungtae Lee	KAIST, South Korea
Jysoo Lee	KISTI, South Korea
Dan Ma	Singapore Management University, Singapore
Steven Miller	Singapore Management University, Singapore
Dirk Neumann	University of Freiburg, Germany

Karsten Oberle	Alcatel-Lucent Bell Labs, Germany
Rajiv Ranjan	University of Melbourne, Australia
Peter Reichl	Telecommunications Research Center Vienna, Austria
Simon See	Sun Microsystems, Singapore
Satoshi Sekiguchi	AIST, Japan
Arunabha Sen	Arizona State University, USA
Katarina Stanoevska	University of St. Gallen, Switzerland
Burkhard Stiller	University of Zurich, Switzerland
Bruno Tuffin	IRISA/INRIA, France
Kurt Vanmechelen	University of Antwerp, Belgium
Dora Varvarigou	National Technical University of Athens, Greece
Daniel Veit	University of Mannheim, Germany
Gabriele von Voigt	University of Hanover, Germany
Christof Weinhardt	University of Karlsruhe, Germany
Stefan Wesner	HLRS, Germany
Phillip Wieder	University of Dortmund, Germany
Ramin Yahyapour	University of Dortmund, Germany
Wolfgang Ziegler	Fraunhofer Institute SCAI, Germany

Steering Committee

Jörn Altmann	Seoul National University, South Korea
Rajkumar Buyya	University of Melbourne, Australia
Thomas Fahringer	University of Innsbruck, Austria
Junseok Hwang	Seoul National University, South Korea
Hing-Yan Lee	National Grid Office, Singapore
Jysoo Lee	KISTI, South Korea
Steven Miller	Singapore Management University, Singapore
Dirk Neumann	University of Freiburg, Germany
Daniel Veit	University of Mannheim, Germany

Sponsoring Institutions

Seoul National University, Seoul, South Korea
University of Cardiff, Cardiff, UK
Springer LNCS, Heidelberg, Germany
Euro-Par 2010, Ischia, Itly

Table of Contents

Session A: Service Evaluation and Trust

Session B: Service Pricing and Software Licenses

Session C: Work in Progress on Adoption of Grid and Cloud Services

Session D: Work in Progress on Value Chains and Service Level Agreements

Technology Transfer of Dynamic IT Outsourcing Requires Security Measures in SLAs

Frank Dickmann[1], Maximilian Brodhun[1], Jürgen Falkner[2], Tobias A. Knoch[3,4], and Ulrich Sax[5]

[1] Department of Medical Informatics, University of Göttingen
Robert-Koch-Straße 40, 37075 Göttingen, Germany
fdickmann@med.uni-goettingen.de, maxi_brodhun@hotmail.com
[2] Fraunhofer-Institute for Industrial Engineering, Nobelstraße 12, 70569 Stuttgart, Germany
juergen.falkner@iao.fraunhofer.de
[3] Biophysical Genomics, Dept. Cell Biology & Genetics, Erasmus MC
Dr. Molewaterplein 50, 3015 GE Rotterdam, Netherlands
[4] Biophysical Genomics, Genome Organization & Function, BioQuant Center / German Cancer Research Center, Im Neuenheimer Feld 267, 69120 Heidelberg, Germany
ta.knoch@taknoch.org
[5] Department of Information Technology, University Medicine Göttingen
Robert-Koch-Straße 40, 37075 Göttingen, Germany
usax@med.uni-goettingen.de

Abstract. For the present efforts in dynamic IT outsourcing environments like Grid or Cloud computing security and trust are ongoing issues. SLAs are a proved remedy to build up trust in outsourcing relations. Therefore, it is necessary to determine whether SLAs can improve trust from the perspective of the outsourcing customer by integration of security measures. The conducted survey indicates that customers see SLAs as an approach to increase their level of trust in IT outsourcing partners. In addition, security measures in SLAs are of high relevance to support trust but not yet integrated appropriately. However, SLAs are very important for the technology transfer of eScience projects in Grid computing. Again, Grid based outsourcing of biomedical IT services requires security measures in SLAs. Thus, the technology transfer process of dynamic IT outsourcing infrastructures requires adequate SLAs in order to be successful.

Keywords: Grid computing, technology transfer, distributed IT outsourcing, Service Level Agreements, security, trust.

1 Introduction

Present efforts towards Grid and Cloud computing infrastructures are, by their very nature, economically oriented efforts for outsourcing of particular tasks to another economic entity which is specialized in the respective field [2, 3]. In fact, the ad hoc character of connecting entities is the difference to previous ways of IT outsourcing e.g. hosting. Accordingly, the outsourcing concepts of Grid and Cloud Computing provide more flexibility in multi-institutional IT integration [4]. In Grid and Cloud

J. Altmann and O.F. Rana (Eds.): GECON 2010, LNCS 6296, pp. 1–15, 2010.

computing environments the customer is intended to receive IT service power on demand [3]. This is to be described in more generic terms as: dynamic IT outsourcing (DITO). Presently, the exact term "dynamic IT outsourcing" is not yet wildly mentioned but definitely hits the heart of the matter [5-7].

A major issue for the service customer in outsourcing IT is trust in the reliability of the service provider [8]. The lack of physical access to outsourced resources is a predominant reason for that trust issue [9]. Besides, trust is a continuous research topic in information management [1, 10] (see also **Fig. 1**). In particular, establishing trust in virtual environments relies on security measures [11].

To determine and negotiate quality of service (QoS) in DITO environments Service Level Agreements (SLAs) are necessary [12]. In addition, SLAs are essential for a successful results by IT outsourcing [13]. The character of DITO implies on demand negotiation of these SLAs. In this context frameworks and mechanisms for automatic SLA management have been developed [14, 15]. Nevertheless, the security issue has not been addressed in this context to a large extent because security measures in SLAs are not integrated yet [16]. Consequently, there is still a lack of trust in dynamic IT outsourcing services. In order to achieve higher economic efficiency by improved resource usage the integration of security measures in SLAs has, however, to be developed [4, 17].

The German D-Grid initiative [18] includes general infrastructure services like accounting or monitoring offered by academic service providers. Unfortunately, SLAs between these service providers and those who offer profession related services are not yet established. For instance, several biomedical Grid services require high security standards due to processing of human identifying data [19]. Such standards include the confidentiality of accounting data as well. Accounting data can support identifying patients when being revealed to a third party under particular circumstances.

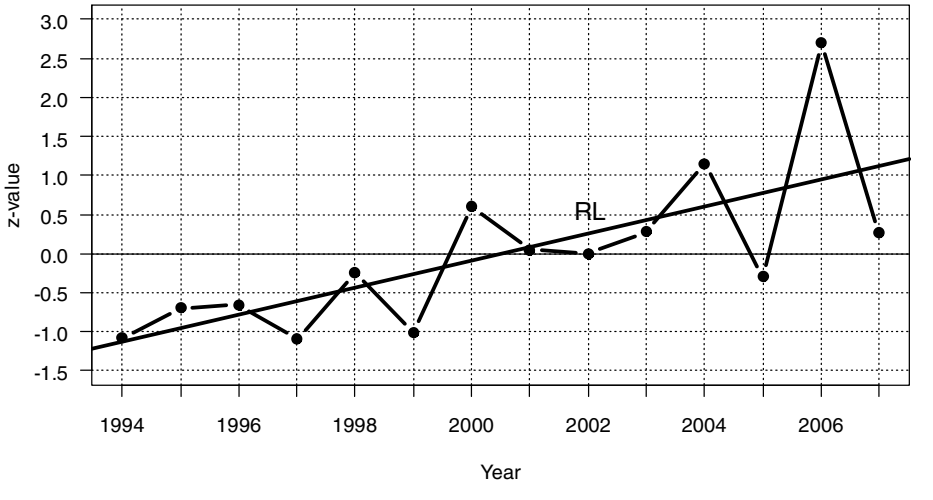


Fig. 1. Trend of the topic "trust" in information systems research: the z-value represents standardized scores (mean value $\mu = 0$, standard deviation $\sigma = 1$) of occurrences of topics in research of information management per year and RL is the linear regression line; line chart adapted from [1]

Thus, the biomedical Grid computing project Services@MediGRID [20] is going to establish a contractual SLA with the D-Grid accounting service provider to achieve a successful technology transfer. Technology transfer means transfer of a scientific result to the "real world" [21]. In essence, successful technology transfer requires trust to be established between biomedical researchers and service providers.

In the given context it is necessary to determine the perception of SLAs by service customers and whether SLAs enable trust in DITO relationships. In addition, the opinion on the topic of security in SLAs is required to be considered. From the introduction the following hypotheses are being derived:

Hypothesis H1: *Security measures in Service Level Agreements positively influence trust supported by Service Level Agreements in dynamic IT outsourcing.*

Hypothesis H2: *Security measures are not yet considered appropriately in Service Level Agreements for dynamic IT outsourcing relationships.*

2 Materials and Methods

In order to get an overview at the opinion of service customers with regard to security measures in SLAs and trust a quantitative survey has been conducted. The primary objective is to verify the hypotheses H1 and H2 as well as adjacent concerns. Therefore, present and potential customers of Cloud or Grid computing services are being queried.

2.1 Choice of Survey Location and Survey Process

As survey location the CeBIT (http://www.cebit.de) 2010 was chosen, due to its attraction to IT experts from all over the world. The survey has been conducted on March, 4th 2010. Additionally, Cloud computing was one of the major topics of the CeBIT exhibition in 2010. This increases the probability of encountering entities of the target group: Cloud computing customers as well as potential Cloud computing customers.

To increase the probability to encounter relevant entities the survey location was restricted to areas near exhibition booths of Cloud computing providers. In these areas randomly picked people have been interviewed. For improved results leisure visitors were excluded. The survey was conducted by two persons whose location was changed in the exhibition area.

2.2 Questionnaire Design

For design of the questionnaire and evaluation of the answers the software GrafStat was applied. GrafStat supports designing questionnaires, collecting answers as well as statistical analysis of the results (http://www.grafstat.de). Further analysis and graphical output of contingency tables was produced with the statistics tool R (http://www.r-project.org).

Achieving results in a direct survey requires a questionnaire which can be completed by the respondents within 2 to 3 minutes. Therefore, a maximum of 12

questions was set. For a quantitative survey multiple-choice answers were used for practical reason. Since Grid and Cloud computing are more commonly known, these terms were chosen as representatives for dynamic IT outsourcing.

Question Q1: *Do you already use Cloud or Grid Computing services? (e.g. Software-as-a-Service, Platform-as-a-Service or Infrastructure-as-a-Service)*

The structure of the questions begins with a general perspective by asking whether the respondent already uses Cloud or Grid computing services or not. This is intended to examine the familiarity with the respective central topic. The survey has an unspecified character with regard to particular occurrences of dynamic IT outsourcing. Thus, all three layers of IT outsourcing are encompassed: 1. Software-as-a-Service (SaaS), 2. Platform-as-a-Service (PaaS), and 3. Infrastructure-as-a-Service (IaaS) [22].

Question Q2: *How much do you trust Cloud or Grid Computing services by external providers?*

Trust as a contemporary issue in dynamic IT outsourcing is queried by a range of high, neutral, and low level of trust in external providers.

Question Q3: *Would you agree that Service Level Agreements support trust in Cloud or Grid Computing providers?*

The question regarding trust in Q2 is followed by the narrowing question whether SLAs support trust in the service of dynamic IT outsourcing providers.

Question Q4: *From your point of view, are Service Level Agreements supporting transparent specifications of costs and services?*

In any case the transparent definition of costs and service parameters is a major concern when defining an SLA. Therefore, this topic is required to be considered.

Question Q5: *Are technical and legal contents of SLA documents you have read clearly described and easy to understand?*

Because the language of SLAs consists of both technical and legal expressions and wording it is necessary to include the apprehension of the respondents. The understanding of the contents depends on the qualification / background as well as clarity. Here both scopes need to be queried separately whether they are clear to the respondent or not.

Question Q6: *Are transparent evaluation and report methods applied by service providers?*

For DITO customers it is important to determine the QoS. In this context the service provider is required to offer a transparent evaluation solution. For decision makers the data produced by the evaluation solution is usually prepared as a report. In order to be able to make profound decisions the quality of the evaluation data is crucial. Consequently, the quality of the QoS evaluation process and its transparency to the customer is a relevant aspect.

Question Q7: *Related to your experience, are Service Level Agreements designed flexibly in order to be adjusted to technical innovations or changes in processes?*

Technological advancement requires adjustment of respective business processes. Accordingly, it is compulsory to design SLAs to be easily adapted to future changes. If an SLA is designed to be adaptive, it will not be necessary to sign a whole new version of the document. Changes will be applied to the existing document by mutual agreement. This reduces administrative efforts.

Question Q8: *Do you have experience in machine-negotiated Service Level Agreements in dynamic infrastructures like Cloud or Grid Computing?*

As mentioned in the introduction, SLAs in dynamic IT outsourcing environments are proposed to be managed by the IT infrastructure itself. Because this is a rather new way of dealing with SLAs, the familiarity with automated SLAs is a relevant question to ask.

Question Q9: *Are machine-negotiated Service Level Agreements able to establish the same level of trust as paper-based contracts?*

For instance, German and U.S. law do not stipulate that a contract needs to be in writing. Contracting relationships, being signed by using IT devices e.g. via click on a URL, are accepted. Nonetheless, trust and written contracts might closely go together. Therefore, it is necessary to evaluate whether automated SLAs can fulfill the same degree of trust.

Question Q10: *How would you measure the relevance of the topic "Security" in Service Level Agreements regarding Cloud or Grid Computing services?*

As mentioned in the introduction, trust can be established by offering adequate security in relation to the according product or service. Security in DITO can be ensured e.g. by specification of data handling procedures or certification of the provider by a neutral institution. Based on the security measures the customer has the ability to assess the Security of Service (SoS). However, SLAs are able to include security measures and to ensure their application. To validate hypothesis H1, a range of high, neutral, and low level of relevance is queried. Because security is a concern regardless of the location of the supplier [23], differences between international and national IT outsourcing is not necessary here.

Question Q11: *From your point of view, is the topic "Security" in Service Level Agreement handled with an adequate amount of importance? (e.g. the definition of preventive measures based on the workflows or risk management processes)*

Addressing the introduced lack of consideration of security in SLAs it is vital to validate hypothesis H2. Security parameters can be addressed via definition of requirements, respective workflows and risk management procedures.

Question Q12: *What do you expect from the use of Cloud or Grid Computing and what advantages do you see for your company?*

After all, it is relevant what opportunities in dynamic IT outsourcing the respondents see for themselves or their company. Since the answers cannot be predicted or should be restricted, the type of answer is defined as free text.

The defined scales in Q2 and Q10 are represented by values: 3 = high, 2 = neutral, 1 = low based on the Likert-scale question style [24]. With the definition of numeric values average and median can be determined. Q5 aims at two different subtopics, the legal and the technical perspective and is therefore implemented as a ordinal-polytomous question type [25]. For Q1, Q3 and Q4, Q6 to Q9 as well as Q11 the dichotomous question type was applied [26].

2.3 Evaluation Methodology

The methodology is based on descriptive statistics in order to analyze the quantitative data. Descriptive statistics encompass the investigated frequencies of occurrence and mean values [27].

In addition, the quantitative data is categorized due to the design of the questionnaire which subdivides the respondents into two major groups. These groups are defined by Q1 as experienced and not experienced DITO customers. Based on contingency tables coherences within the data were analyzed [28]. Moreover, mosaic diagrams are used for optimal representation of the contingency table results [29].

3 Results

The results are described in two parts. The first part outlines the general results from the questions. In the second part relevant cross reference results are presented.

3.1 General Survey Responses

Within the survey 75 questionnaires were filled out in total. A majority of 57.3 % of the participants already uses Cloud or Grid computing services (see Q1 in **Table 1**). Furthermore, 36.0 % of the respondents show high confidence in external DITO providers. Further 46.7 % express a neutral and 16.0 % a low degree of trust (see Q2 in **Table 2**). On the scale from 1 (low) to 3 (high) the average degree of trust equals 2.20 according to the 74 received answers. This indicates that a majority of the participants trust IT outsourcing partners.

A positive coherence between SLAs and trust in IT outsourcing relationships is confirmed by a major share of 73.3 % of the participants (see Q3 in **Table 1**). This supports the initial assumption. Moreover, 50.7 % positive results indicate that SLAs are implemented with transparent cost and service definitions (see Q4 in **Table 1**). Still yet, 38.7 % of the replies show the opposite. Consequently, transparency is still an issue but not a major one.

With regard to the clarity of SLA contents 29.3 % of the respondents are positive about the legal details while 46.7 % do not share the same opinion (see Q5 in **Table 3**). According to technical details, 54.7 % are positive about the technical contents while 22.7 % are not. The results evidently show a lack of legal clarity to the customers in SLA documents. Unfortunately, Q5 was not answered 12 times and 11 times partially responded. From the 11 partial answers, 6 legal and 5 technical answers have been skipped.

Transparent evaluation measures and reports are offered by service providers in the opinion of 44.0 % of the interviewed CeBIT visitors whereas 42.7 % object (see Q6 in **Table 1**). This further backs up the determined lack of transparency indicated by Q4. An adaptive design of SLAs is confirmed by 45.3 % of the respondents while 44.0 % do not share the same opinion (see Q7 in **Table 1**).

Experience in machine-negotiated SLAs state 30.7 % of the participants whereas a majority of 66.7 % does not (see Q8 in Table 1). Subsequently, 40.0 % of the interviewed CeBIT visitors trust automatically negotiated SLAs with the same level as written ones (see Q9 in Table 1). Besides, 50.7 % of the interviewed visitors do not share that trust. These results generally indicate that automatic SLAs are not perceived as to be a respected instrument. The central question concerning the topic of security in SLAs was skipped by 3 respondents (see Q10 in **Table 2**). 72.0 % of the answers show a high, 20.0 % neutral, and 4.0 % low relevance for security measures to be covered by SLAs. This supports hypothesis H1. According to hypothesis H2, 26.7 % of the responses agree whereas 58.7 % disagree whether the topic of security is being addressed appropriately in SLAs (see Q11 in **Table 1**).

Table 1. The Results Overview for the Dichotomous Questions

Question	Answer	Yes	No	Skipped
Q1:	Do you already use Cloud or Grid Computing services?	43 (57.3 %)	32 (42.7 %)	0 (0.0 %)
Q3:	Would you agree that Service Level Agreements support trust in Cloud or Grid Computing providers?	55 (73.3 %)	17 (22.7 %)	3 (4.0 %)
Q4:	From your point of view, are Service Level Agreements supporting transparent specifications of costs and services?	38 (50.7 %)	29 (38.7 %)	8 (10.6 %)
Q6:	Are transparent evaluation and report methods applied by service providers?	33 (44.0 %)	32 (42.7 %)	10 (13.3 %)
Q7:	Related to your experience, are Service Level Agreements designed flexibly in order to be adjusted to technical innovations or changes in processes?	34 (45.3 %)	33 (44.0 %)	8 (10.7 %)
Q8:	Do you have experience in machine-negotiated Service Level Agreements in dynamic infrastructures like Cloud or Grid Computing?	23 (30.7 %)	50 (66.7 %)	2 (2.6 %)
Q9:	Are machine-negotiated Service Level Agreements able to establish the same level of trust as paper-based contracts?	30 (40.0 %)	38 (50.7 %)	7 (9.3 %)
Q11:	From your point of view, is the topic "Security" in Service Level Agreement handled with an adequate amount of importance?	20 (26.7 %)	44 (58.7 %)	11 (14.6 %)

Table 2. The Results Overview for the Likert-Scale Questions

Question	Answer	High	Neutral	Low	Skipped
Q2:	How much do you trust Cloud or Grid Computing services by external providers?	27 (36.0 %)	35 (46.7 %)	12 (16.0 %)	1 (1.3 %)
Q10:	How would you measure the relevance of the topic "Security" in Service Level Agreements regarding Cloud or Grid Computing services?	54 (72.0 %)	15 (20.0 %)	3 (4.0 %)	3 (4.0 %)

Table 3. The Results Overview for the Ordinal-Polytomous Question

Question	Answer	Clear	Unclear	Skipped
Q5:	Are technical and legal contents of SLA documents you have read clearly described and easy to understand?			
	Legally	22 (29.3 %)	35 (46.7 %)	18 (24.0 %)
	Technically	41 (54.7 %)	17 (22.7 %)	17 (22.6 %)

According to expectations from Grid or Cloud computing by the participants in Q12, the question was answered 36 times. Because of free text, only the most frequently given answers are documented. With 7 times “lower costs” was the most frequent answer followed by 6 times for “higher security / availability”. Additional 4 respondents expect “more flexibility / scalability”. Table 1, 2, and 3 summarize the overall results. All percentages are rounded up to one place behind the decimal point.

3.2 Contingency Table Results

For further evaluation it is presumed that participants with DITO experience (see Q1 in **Table 1**) are supposed give more reliable answers. In addition, skipped answers are no further included.

Considering trust, 48.8 % of the DITO users have high, 34.9 % neutral, and 16.3 % low confidence in external service providers (coherence: Q1, Q2; see **Table 4**). The mean value equals 2.20 and shows no difference in comparison to the whole population. Besides, from 42 experienced users 78.6 % of the respondents agree that SLAs support trust in DITO relationships (coherence: Q1, Q3). Here, no significant difference exists compared to inexperienced users. A positive coherence exists between Q2 and Q3. A majority of the experienced users have high or neutral confidence in external providers as well as support that SLAs support such trust in external providers (see **Fig. 2**).

Additionally, 23 (57.5 %) from 40 skillful participants agree that SLAs provide a transparent specification of costs and services (coherence: Q1, Q4). Anyhow, the understanding of SLAs by experienced users is more technically oriented (coherence of Q1 and Q5). Furthermore, from 41 experienced users 23 (56.1 %) respond that

transparent evaluation and report methods are applied by service providers (coherence: Q1, Q6). Considering the given 40 responses to adaptability of SLAs, 25 (62.5 %) experienced users agree (coherence: Q1, Q7).

Moreover, from 42 experienced DITO users 27 (64.3 %) are not familiar with machine-negotiated SLAs. In contrast to the general results 11 (73.3 %) of 15 who have experience with machine-negotiated SLAs and DITO state that these SLAs offer the same level of trust as their written counterparts (coherence: Q1, Q8, Q9; see **Fig. 3**). Therefore, machine-negotiated SLAs are mostly perceived as adequate if applied.

With regard to the importance of security in SLAs 36 (85.7 %) responses show high, 5 (11.9 %) neutral, and 1 (2.4 %) low relevance (coherence: Q1, Q10). Thus, hypothesis H1 is assumed to be valid.

For validation of hypothesis H2 the importance for security in SLAs and its adequate amount of consideration are relevant (coherence: Q10, Q11). Here, 64 answers are given to both questions in total. A majority of 38 (59.4 %) respondents do not agree that security is considered in SLAs appropriately while at the same time they see high relevance for security in SLAs. In particular, a more significant majority of 27 (69.2 %) of the 39 experienced respondents support that statement (coherence: Q1, Q10, Q11; see **Table 5** and **Fig. 4**). Thus, hypothesis H2 is assumed to be valid.

Table 4. Coherence between user experience and confidence in external service providers

		Q1: Do you already use Cloud or Grid Computing services?	
		Yes	**No**
Q2: How much do you trust Cloud or Grid Computing services by external providers?	**High**	21 (48.8 %)	5 (16.1 %)
	Neutral	15 (34.9 %)	20 (65.5 %)
	Low	7 (16.3 %)	6 (19.4 %)
Total		43 (100.0 %)	31 (100.0 %)

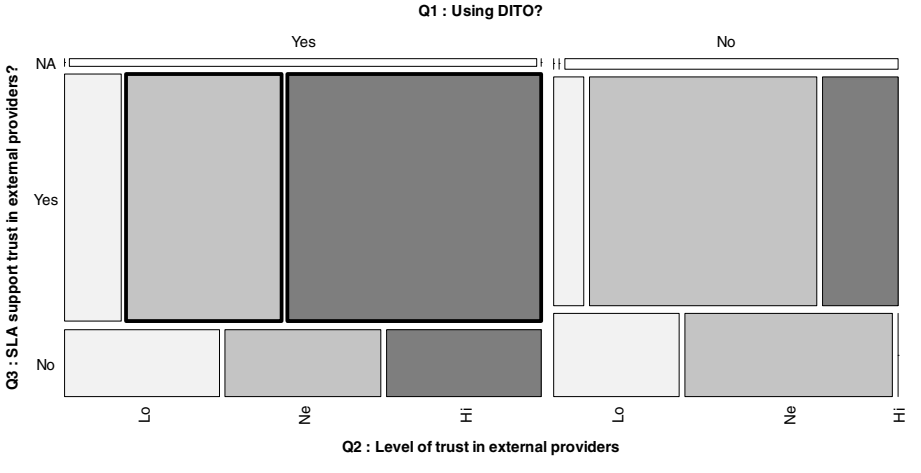


Fig. 2. Coherence between User Experience, SLAs Supporting Trust and the Level of Trust in External Providers

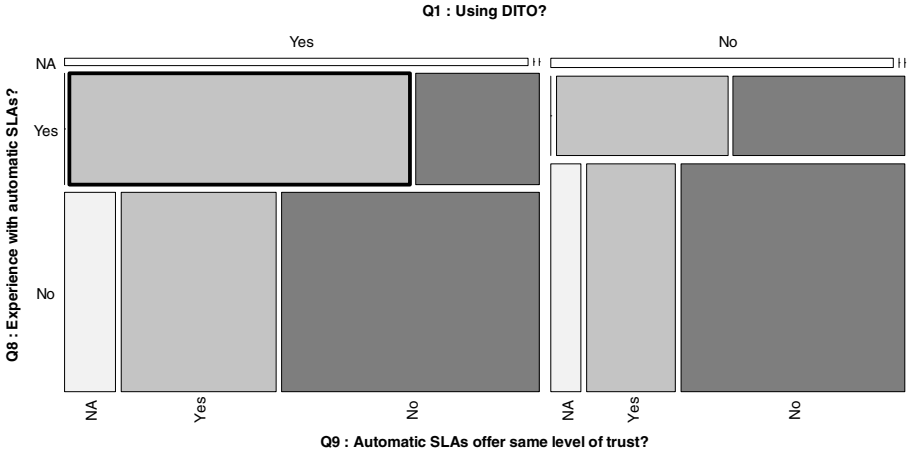


Fig. 3. Coherence between user experience, experience and trust in machine-negotiated SLAs

Table 5. Coherence between user experience, security consideration and relevance to SLAs

				Q1: Do you already use Cloud or Grid Computing services? Yes	No
Q11: Is security handled with the adequate amount in SLAs?	no	Q10: Relevance of security in SLAs	High	27 (69.2 %)	11 (44.0 %)
			Neutral	1 (2.6 %)	3 (12.0 %)
			Low	0 (0.0 %)	2 (8.0 %)
	yes	Q10: Relevance of security in SLAs	High	7 (17.9 %)	4 (16.0 %)
			Neutral	3 (7.7 %)	5 (20.0 %)
			Low	1 (2.6 %)	0 (0.0 %)
Total				39 (100.0 %)	25 (100.0 %)

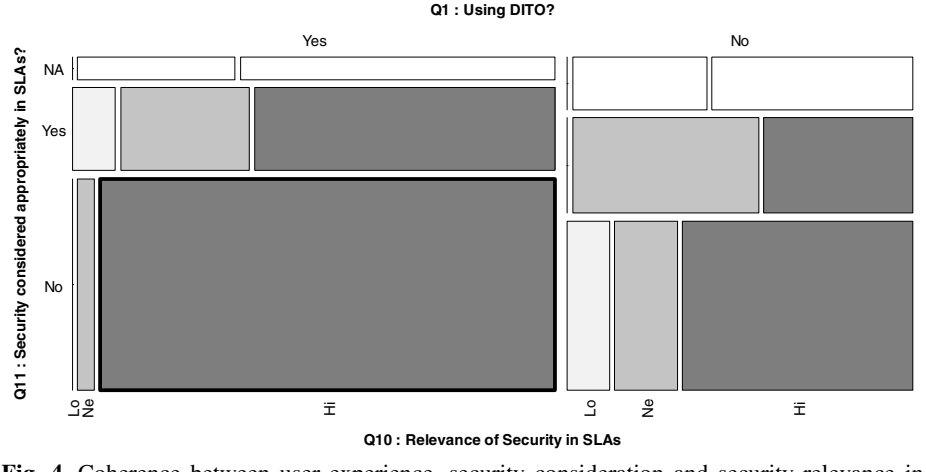


Fig. 4. Coherence between user experience, security consideration and security relevance in SLAs

4 Discussion and Outlook

The relatively small number of participants in this survey does not necessarily offer a fully representative conclusion. Nevertheless, the clear results – which are also profoundly supported by the experienced share of the participants – can be assumed to be a good reference to reality.

Because a clear majority of the randomly interviewed survey participants already uses Grid or Cloud computing services indicates that distributed IT outsourcing is an important matter. Nonetheless, there is still potential to acquire new DITO customers. More clearly, as one of the respondents stated: "It is the natural path (Isaac Asimov)".

The overall high trust in DITO providers opposes present discussions on data security in social network services [30]. On the other hand, it sustains that users have strong trust in certain social network services [31]. Another example for concerns about data stored by external institutions is related to scientific data archives for long term preservation and / or open access. Researchers mostly do not like the idea to store content in data sharing environments because someone else could achieve something more significant with their data [32]. Such concerns affect possible developments on Grid and Cloud computing in the academic area and consequently the technology transfer process of Grid computing. Building up trust via security in SLAs will therefore unquestionably support prevalence of use of DITO services.

The aim of the German D-Grid initiative and of the project Services@MediGRID is a technology transfer from research and development to a commercially oriented level. Commercial – a service being paid for – also includes, in this case, the utilization of the D-Grid infrastructure as a product in the academic area. In the academic area DITO will be provided by academic IT service providers. However, as explained, SLAs are necessary as a substantial basis for the service provision. Hence, the further discussion will emphasize the requirements for SLAs by taking technology transfer into account.

4.1 Requirements for Service Level Agreements

Generally, hypotheses H1 and H2 are assumed to be validated by the conducted survey. Thus, security measures in SLA are important to establish trust between customers and providers of dynamic IT outsourcing. Nonetheless, those security measures are not yet embedded in SLAs. Figure 4 precisely refers to this matter. Similar experiences within the project Services@MediGRID show that the technology transfer lacks because SLAs in general and security measures covered by SLAs are still missing. For that reason the process of technology transfer, at least partly, depends on profound contractual conditions. In case of the biomedical area this necessarily encompasses security measures to be defined.

Though a majority of experienced users trusts automatic SLAs overall, written contractual agreements are still preferred over machine-negotiated alternatives. This opinion is likewise supported by the inexperienced participants. It would be important to know the particular reasons for that matter, but that was not an objective of the survey. However, a significant number of respondents have a rather technical perspective according to Q5 (see **Table 3**). Because of the technical focus the qualification of the respondents does not sustain their judgment that automatic SLAs offer the same

level of trust as their written counterparts. Notwithstanding, Grid science projects like D-Grid rely on the development of automatic SLAs. In essence, it can be deducted that the lack of consideration of written contracts in scientific Grid and Cloud computing projects interferes with successful technology transfer. Above all, it can be stated that automatic negotiated SLAs require to be covered by additional written general SLAs. Even though automatically negotiated SLAs are necessary for efficiently providing ad hoc IT services, a holistic approach is considered to include a general agreement as well. This backs up the initiative of Services@MediGRID to establish the use of contractual agreements.

Further, more general, requirements for SLAs can be derived from the overall results of section 3.1. This encompasses the common necessity of transparency which includes clear specification of costs and products – respectively of the DITO services. According to the results this aspect is already adequately addressed. Yet, transparent evaluation methods like reports still offer some potential for optimization. In the same way follows the adaptability of SLAs.

4.2 Outlook

Since Grid and Cloud computing as specific characteristics of DITO environments are relatively new, the technology transfer process has not been entirely completed in areas like biomedicine yet. With focus on building up trust by including security measures in SLAs it can be stated that present SLAs do not support that matter to a sufficient degree at present. This is an important indication because SLAs are necessary to build up trust in dynamic IT outsourcing. Besides, definition of the security measures is required as well.

It can also be concluded that customers of IT services still rely on good old paper rather than on electronic agreements. This is required to be considered when it comes to implementing Grid or Cloud computing services successfully. Economic optimization, in the first place, does not include optimization of trust. Hence, the relevant steps toward trust have to be laid out in further research. First and foremost, in the area of biomedicine security in SLAs is necessary. Moreover, the technology transfer process of Grid and Cloud computing depends on obligatory promises by service providers. SLAs as an instrument for implementing reliability in outsourcing relationships are thus required for technology transfer in the IT context.

An interesting indication was given by 12 respondents. They stated that SLAs are technically as well as legally clear to them. Further questions revealed that these persons were originally coming from the field of law and gained additional knowledge in IT or are now working in a primarily IT related context.

Due to the fact that D-Grid has not yet established sustainable contractual relationships, the commercial IT association in Germany BITKOM has seized the opportunity by proposing the implementation of a German Cloud to the German federal government [33]. This might also lead to a major shift for support with IT resources in the academic field. Accordingly, future efforts of technology transfer of IT infrastructures have to be quicker in establishing general contractual conditions in order to be successful.

Acknowledgements

This publication was supported by the joint project Services@MediGRID funded by the German Federal Ministry of Education and Research (BMBF), FKZ 01IG07015A-G and the joint project WissGrid funded by the German Federal Ministry of Education and Research (BMBF), FKZ 01IG09005A-L.

References

1. Steininger, K., Riedl, R., Roithmayr, F., Mertens, P.: Fads and Trends in Business and Information Systems Engineering and Information Systems Research – A Comparative Literature Analysis. Business & Information Systems Engineering 1(6), 411–428 (2009)
2. BEinGRID Consortium, Approaching the Cloud: Better Business Using Grid Solutions (2009), http://www.beingrid.eu/fileadmin/beingrid/pr_folder/CaseStudy2/BEinGRID_CaseStudies_Booklet_Quotes_VF_LowWeb_012010.pdf (accessed: 2010.02.14)
3. Matros, R., Stute, P., Zuydtwyck, N.H.v., Eymann, T.: Make-or-Buy im Cloud-Computing – Ein entscheidungsorientiertes Modell für den Bezug von Amazon Web Services. Universität Bayreuth, Lehrstuhl für Wirtschaftsinformatik, Bayreuth, Bayreuther Arbeitspapiere zur Wirtschaftsinformatik (2009), http://opus.ub.uni-bayreuth.de/volltexte/2009/552/pdf/Paper_45.pdf (accessed: 2009.11.15)
4. Dickmann, F., Kaspar, M., Lohnhardt, B., Knoch, T.A., Sax, U.: Perspectives of MediGRID. Stud. Health Technol. Inform. 147, 173–182 (2009)
5. Google Scholar: Results for dynamic IT outsourcing, http://scholar.google.de/scholar?q=%22dynamic+it+outsourcing%22&hl=de&btnG=Suche&lr=
6. Jain, A.: Towards A Systemic View Of Organizational Dynamic IT Capability: An Empirical Assessment. University of Texas at Arlington, Arlington (2007), http://hdl.handle.net/10106/238 (accessed: 2010.02.01)
7. Vassiliadis, B., Stefani, A., Tsaknakis, J., Tsakalidis, A.: From application service provision to service-oriented computing: A study of the IT outsourcing evolution. Telematics and Informatics 23(4), 271–293 (2006)
8. Malik, Z., Bouguettaya, A.: Trust Management for Service-Oriented Environments. Springer, Heidelberg (2009)
9. Talbot, D.: Security in the Ether. Technology Review 50(1), 36–42 (2010)
10. Arenas, A., Wilson, M., Matthews, B.: On trust management in grids. In: Davide, F. (ed.) 1st International Conference on Autonomic Computing and Communication Systems, Autonomics 2007, pp. 1–7. ACM, Rome (2007)
11. Yau, P.-W., Tomlinson, A.: Using Trusted Computing in Commercial Grids. In: Akhgar, B. (ed.) International Workshops on Conceptual Structures, ICCS 2007, pp. 31–36. Springer, Sheffield (2007)

12. Buyya, R., Yeo, C.S., Venugopal, S.: Market-Oriented Cloud Computing: Vision, Hype, and Reality for Delivering IT Services as Computing Utilities. The Computing Research Repository (CoRR) abs/0808.3558 (2008)
13. Goo, J., Nam, K.: Contract as a Source of Trust–Commitment in Successful IT Outsourcing Relationship: An Empirical Study. In: 40th Annual Hawaii International Conference on System Sciences (HICSS 2007), p. 239a. IEEE Computer Society, Waikoloa (2007)
14. Brandic, I., Music, D., Leitner, P., Dustdar, S.: VieSLAF Framework: Enabling Adaptive and Versatile SLA-Management. In: Altmann, J., Buyya, R., Rana, O.F. (eds.) GECON 2009. LNCS, vol. 5745, pp. 60–73. Springer, Heidelberg (2009)
15. Padgett, J., Haji, M., Djemame, K.: SLA Management in a Service Oriented Architecture. In: Gervasi, O., Gavrilova, M.L., Kumar, V., Laganá, A., Lee, H.P., Mun, Y., Taniar, D., Tan, C.J.K. (eds.) ICCSA 2005. LNCS, vol. 3483, pp. 1282–1291. Springer, Heidelberg (2005)
16. Parrilli, D.M.: Legal Issues in Grid and Cloud Computing. In: Stanoevska-Slabeva, K., Wozniak, T., Ristol, S. (eds.) Grid and Cloud Computing, pp. 97–118. Springer, Heidelberg (2009)
17. Knoch, T.A., Baumgartner, V., de Zeeuw, L.V., Grosveld, F.G., Egger, K.: e-Human Grid Ecology - understanding and approaching the inverse tragedy of the commons in the e-Grid society. Stud. Health Technol. Inform 147, 269–276 (2009)
18. D-Grid Initiative, http://www.d-grid.de
19. Krefting, D., Bart, J., Beronov, K., Dzhimova, O., Falkner, J., Hartung, M., Hoheisel, A., Knoch, T.A., Lingner, T., et al.: MediGRID: Towards a user friendly secured grid infrastructure. Future Generation Computer Systems 25(3), 326–336 (2009)
20. Services@MediGRID, http://services.medigrid.de
21. Bozeman, B.: Technology transfer and public policy: a review of research and theory. Research Policy 29(4-5), 627-655 (2000)
22. Stanoevska-Slabeva, K., Wozniak, T.: Cloud Basics – An Introduction to Cloud Computing. In: Stanoevska-Slabeva, K., Wozniak, T., Ristol, S. (eds.) Grid and Cloud Computing, pp. 47–62. Springer, Heidelberg (2009)
23. Burnett, R.: Outsourcing IT: the legal aspects: planning, contracting, managing and the law. Gower Publishing Ltd.(2009)
24. Trochim, W.M.K.: Likert Scaling. In: Research Methods Knowledge Base (2006), http://www.socialresearchmethods.net/kb/scallik.php (accessed: 2010.04.08)
25. Wikipedia contributors, Statistical survey. Wikipedia, The Free Encyclopedia (2010), http://en.wikipedia.org/wiki/Statistical_survey
26. Adèr, H.J., Mellenbergh, G.J., Hand, D.J.: Advising on research methods: a consultant's companion. Van Kessel, Huizen (2008)
27. Eckle-Kohler, J., Kohler, M.: Eine Einführung in die Statistik und ihre Anwendungen. Springer, Berlin (2009)
28. Mills, A.J., Durepos, G., Wiebe, E.: Encyclopedia of case study research. SAGE Publications, Thousand Oaks (2010)
29. Friendly, M.: Mosaic Displays for Multi-Way Contingency Tables. Journal of the American Statistical Association 89(425), 190–200 (1994)
30. Lohr, S.: How Privacy Vanishes Online. The New York Times, New York (2010), http://www.nytimes.com/2010/03/17/technology/17privacy.html

31. Bonneau, J., Preibusch, S.: The Privacy Jungle: On the Market for Data Protection in Social Networks. In: The Eighth Workshop on the Economics of Information Security (WEIS 2009), London, UK (2009)
32. Nelson, B.: Data sharing: Empty archives. Nature 461(7261), 160–163 (2009)
33. Briegleb, V.: Deutsche IT-Branche will die Cloud mitgestalten. In: Heise online, Heise Verlag, Hannover (2010), http://www.heise.de/ix/meldung/Deutsche-IT-Branche-will-die-Cloud-mitgestalten-953839.html (accessed: 2010.03.12)

Service Selection Decision Support in the Internet of Services

Konstantinos Tserpes, Fotis Aisopos, Dimosthenis Kyriazis, and Theodora Varvarigou

Dept. of Electrical and Computer Engineering, National Technical University of Athens, 9, Heroon Polytechniou Str, 15773 Athens, Greece
{tserpes,fotaisopos,dkyr,dora}@telecom.ntua.gr

Abstract. With the emergence of service provisioning infrastructures and the networking capabilities of the web the IT world has been enhanced with the dynamics to support a market of services, where the number of providers and customers is potentially unbounded, also known as Internet of Services. In that frame, the customer comes against the problem of selecting a service from a plethora of available ones. This paper advocates that the large number of customers can provide the solution to this problem, by exploiting the experience of those that present similar behavior when they are asked to rate a provided service.

Keywords: Internet of Services, Service Oriented Architectures, decision support, Service Selection.

1 Introduction

The emergence of the Internet of Services concept is expected to lead to a deconstruction of the application services as they are already delivered in existing "closed" service provisioning environments. The involvement of all kinds of players, from major service providers to home users in the provisioning process is anticipated to create an open market of services. In such a market, demand and offer will define the complexity of the services that will comprise the baseline, core functionalities that once put together will constitute greater workflows. Hence, the distinction between providers and customers will not be as clear as it is now, creating a major need to incorporate various criteria in the service selection process. These criteria must be able to express each individual customer's requirements in a market where services with similar functionality may be provided by a vast amount of providers, with different configurations, dependencies, prices and terms. We propose a model and a mechanism to provide recommendations to customers for invoking services in an unknown to them market using collaborative filtering techniques.

The proposed concept lies on the foundations of existing service provisioning systems and especially Service Oriented Architectures. Service Level Agreement (SLA) is a fundamental tool for governing the customer-provider relationship, defining and quantifying the required levels of the Quality of Service (QoS). A number of terms

J. Altmann and O.F. Rana (Eds.): GECON 2010, LNCS 6296, pp. 16–33, 2010.

defined based on the particular application are included in the SLA "document" so as to enable the customer to specify the service needs based on the provider specific capabilities. Upon agreement, the customer can monitor the delivery of QoS and once a constraint is violated, the contract is considered breached and the provider is bound to the compensation clauses also included in the SLA.

However, the monitoring of the provider side is not always easy to happen in an objective way (the provider can always lie in order to avoid compensations, e.g. about the amount of resources used). The customer has to usually rely on his own experience as a domain expert in order to measure the delivered quality (e.g. service reliability). This is usually referred to as Quality of user Experience (QoE) and even though it may have no impact on the actual SLA, it has a direct impact on the assessment of the provider reputation from the customer.

This paper exploits the concept of the QoE in order to allow for customers to evaluate providers' services and use these evaluations for giving recommendations to customers with similar experiences. Using the evaluations the customers are grouped according to their similarity so as to be able to predict their assessment of quality for services that they haven't used so far. Correlating these ratings from the various existing customers it is possible to provide recommendations from customers with similar ratings to others so as to help them in the service selection process.

The proposed service selection model is based on collaborative filtering techniques and especially on measuring the correlation between customer ratings using the Pearson's product-moment correlation coefficient. In order to evaluate the technique, this paper presents a simulation of the model in a service provisioning environment where various datasets are used as evaluation sets.

This document is organized as such: Section 1 is the current section and it serves the purpose of the introductory to the concept paragraph. Section 2 explains the Evaluation Prediction Model by first setting the scene and describing the problem formulation and presents the method used to exploit the current problem. Section 3 is dedicated to the evaluation of the experiment results, using datasets extracted from the real world. Section 4 presents the related work and Section 5 presents the conclusions of the current work.

2 Evaluation Prediction Model

The concept of this work is based on the assumption that in a set of customers with a potentially very large population, it is possible to recognize similarities in the behaviour of distinct groups. This allows for the formation of subgroups that can be examined locally and recognize a generic, single strategy that characterizes the behaviour of each one of them. In this way it becomes possible to predict a single customer's behaviour by identifying the group's strategy.

This kind of service marketplaces has not been formulated clearly yet, however, there are some emerging Cloud computing paradigms, which describe just the case. Infrastructure as a Service (IaaS), Software as a Service (SaaS) and mainly Platform as a Service (PaaS) which are the main Cloud enablers, are existing environments where the borderline between customer and provider is blurred, since anyone can

have access to easy-to-use, cheap and physically resource-less applications (the presence of resources is transparent to the customer). The capability to deliver and receive services easily and cost-efficiently implies itself that a large number of customers and providers exist, creating a market with many players and groups of similar strategies. Typical examples and proof of concept are successful Cloud Computing cases such as Google's App Engine® and Amazon's Web Services®.

In such a service marketplace, a decision support mechanism for service selection can operate in a similar way that the e-Bay's feedback system operates. A customer is relying on the other customers' opinion when they want to invoke a service. The main difference is that in this case the customer doesn't know the service's rating in order to decide but instead the client application service is collecting the ratings and uses them in order to calculate a prediction on behalf of the customer. This role can be easily played by a broker acting as a trusted third party. This role is very important because it helps in avoiding a very basic feedback system shortcoming: that the design of the rating policy influences both the level of trust and efficiency of an e-Bay-like marketplace and therefore systems where ratings are kept secret serve to limit strategic ratings and are proved more effective in enhancing trust [1].

The described model forms the basis for setting up a mechanism that can act as a recommender system. It is based on memory-based collaborative filtering techniques and especially on correlation methods [2]. Similar techniques are vector similarity, inverse user frequency and case amplification [3].

Using correlation methods for recommender systems is not a new idea. In general, since the wide spread of the Internet and the blooming of the electronic commerce, collaborative filtering algorithms were used a lot as recommender systems. Correlation techniques however, stand first amongst the choices of the engineers for the development of decision support systems, as they are reliable, fast and easy to be implemented.

This decision support for service selection technique is implemented as a supporting mechanism on an actual environment that enables service provisioning on a large scale. The environment is based on the architecture principles described in [4], i.e. an SLA-driven service provisioning environment with strong business orientation. Its implementation is the result of the integration of Service Oriented Architecture (SOA) and Grid technologies. In particular, the environment is built using the GRIA middleware ([5], [6]) in which application services can be made available for accessing from the customers through special interfaces. GRIA resolves the resource allocation issues by distributing tasks within the available resources (be it homogeneous or heterogeneous). More importantly, it deals with the non-functional requirements such as trust and security and SLA management using sophisticated mechanisms on the provider as well as on the customer side. GRIA is one of the first Grid middlewares that adopted service orientation as a methodology to resolve interoperability and efficiency issues. Even though, GRIA was selected as the reference architecture, the general concept, is middleware independent and could be implemented in various SOA environments.

Having the above-mentioned in mind an overview of the reference provisioning environment is depicted in Figure 1. The architecture here presents the implementation of the Recommender Service supporting GRIA customers, however no actual dependencies exist between the external Service and the client software, and thus the respective architecture would be efficient for other suchlike SOAs, other than GRIA.

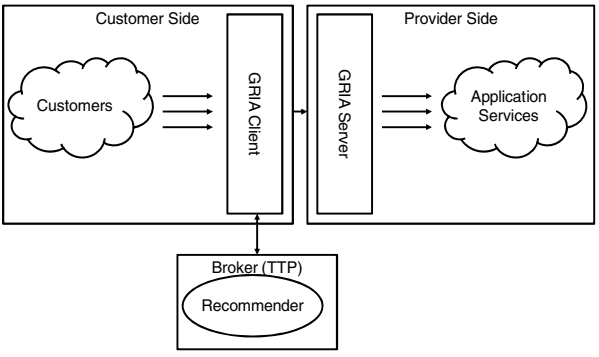


Fig. 1. Overview of the Reference Architecture

In the system depicted in Figure 1, the customers use specific interfaces that are provided by the GRIA Client software. The GRIA Client offers specific interfaces so as to master the whole lifecycle of a job submission, that is, service discovery and selection as well as negotiation and service invocation for the customer side. It also offers a GUI for the user but it can also be accessible through a web portal or any other kind of GUI. The service selection process can be supported by a recommender system through the appropriate port interfaces of the GRIA Client. However, as explained in the previous Section, the recommender system is based on collaborative filtering techniques, which implies the need of an aggregation point of the data coming from various customers. This role is played by a broker who essentially collects all the customer data that are needed in order to proceed with the recommendation. This will be further explained in the following Sections.

Finally, once the selection of application services takes place and the workflow is created, the GRIA Server handles the job requests by assigning the jobs to the various resources that exist in the underlying infrastructure.

The abovementioned process is explained in brief, as in practice various security, trust and SLA management issues are involved but are regarded as out of scope for this document. The focus of the paper is to depict how a recommender system could offer decision making support using collaborative filtering techniques. In what follows, the problem is set in its theoretic dimension so as to better depict the whole evaluation framework that follows and most importantly, the collaboration filtering technique that it is proposed.

The following Section presents the problem formulation that will help the reader to better understand the evaluation framework that will be presented later.

2.1 Problem Formulation

A set of l customers of a service provisioning environment is considered, each one denoted as C_k where $k \in [1, l]$. Each one of them can use a set of n available services, indicated as S_{ki}, where $i \in [1, n]$ the service index. For each service, the

customer and the provider share an SLA which defines *m* quality parameters. For every customer and service these parameters are annotated as P_{kij}, where $j \in [1, m]$.

Each time a service (S_{ai}) is invoked by a customer, say C_a, the customer evaluates the provided quality so as to create a history of evaluations that will be used in the future during the service selection process. This evaluation takes place by simply assigning a rate which indicates his satisfaction in each of the SLA terms. Therefore the evaluation of *m* parameters of a service S_{ai} can be depicted as a vector of ratings: $E_{ai} = [r_1, r_2, \ldots, r_m]$, where r_q depicts the rating of the *q*-th parameter of the SLA from C_a. In the case when more evaluations have been produced for S_{ai} from C_a, the mean values of the ratings are considered as the elements of the evaluation vector: $E_{ai} = [\bar{r}_1, \bar{r}_2, \ldots, \bar{r}_m]$.

Now, let's suppose that two customers C_a and C_u have used at least one common service, which means that: $S_a \cap S_u = I \neq \varnothing$. Therefore, for each common service the evaluations can be compared so as to understand how similar the requirements of the customers are as well as how similar are their satisfactory levels. This is very helpful, because if two customers have high similarity probability for each element of I then they will probably rate at the same way the services that exist in I' ($I \cap I' = \varnothing$). The more customers with which a third customer has similar ratings, the better chance there is to predict the ratings for his respective I'. Having as a reference one customer (say C_a) for whom we would like to predict the rating for a specific service the next three steps are followed:

1. Identify how similar his evaluation vectors are with the rest of the customers that have used the same services, by calculating the correlation coefficient separately for every element of the evaluation vector. This will indicate the strength and direction of the linear relationship between a specific customer and all the others.
2. For those customers that have the highest similarity with the active customer (neighbourhood), compute a prediction of his rating for a specific parameter from a weighted combination of the selected neighbours' ratings.
3. Iterate steps 1) and 2) for every parameter of the service in question for the customer in question.

In the first step, similarity between two customers is computed using the Pearson correlation coefficient for every SLA parameter, defined below:

$$c_{au} = \frac{\sum_{i=1}^{h} (r_{ai} - \bar{r}_a) \times (r_{ui} - \bar{r}_u)}{\sqrt{\sum_{i=1}^{h} (r_{ai} - \bar{r}_a)^2 \times \sum_{i=1}^{h} (r_{ui} - \bar{r}_u)^2}}, \quad (1)$$

where $h = |I_{au}|$, the amount of services evaluated by both C_a and C_u (for readability purposes the set of the services that have been evaluated by users C_a and C_u, are denoted as I_{au} and the complementary set as I'_{au}), r_{ai} is C_a's rate for a specific SLA term of service S_i and $\bar{r}_a = average(r_{a1j}, r_{a1j}, \ldots, r_{ahj})$ is the mean rating of user C_a for a specific SLA term *j* of the services in I_{au}.

Pearson correlation co-efficient determines the extent to which values of two variables are "proportional" to each other. In case the value is 0 then there is no linear relation between the two variables.

Having calculated the correlation coefficient c_{au} for every parameter between some customers, predictions for this parameter are computed as the weighted average of deviations from the other customers' mean:

$$p_{ai} = \bar{r}_a + \frac{\sum_{u=1}^{g} (r_{ui} - \bar{r}_a) \times c_{au}}{\sum_{u=1}^{g} c_{au}} \tag{2}$$

where *g* is the number of customers in the neighbourhood of C_a having used S_{ai} and p_{ai} is the predicted rating of a parameter of S_{ai} for customer C_a. This prediction can also be denoted as p_{aij}, where in this case *j* is the index of the parameters of service S_{ai} (therefore its upper threshold changes dynamically).

The procedure above is iterated for every customer C_u and every parameter of the service S_i in question in order to calculate all the ratings. The result is a vector such as the following:

$$PR_{ai} = \begin{bmatrix} p_{ai1} \\ p_{ai2} \\ \vdots \\ p_{aig} \end{bmatrix} \tag{3}$$

This vector predicts the ratings of a customer for a particular service with predefined quality requirements included in the SLAs. By comparing the various *PR* vectors for services of the same service type according to the customer's business policy one can conclude as to which service instance to use. The customer's business policy may dictate that emphasis on a specific SLA term must be given or that the service that "wins" is the one with the more high valued SLA terms. In order to express this using the presented problem formulation the various vectors are needed to be compared, weighting each element *e* (SLA term rating), using a weight w_e, in order to yield the importance of each SLA term according to the customer preferences. The comparison at that level is taking place using a reference vector, the optimal rating vector R_{opt}. R_{opt} represents the case where the customer is fully satisfied, that is, the elements of R_{opt} take the maximum possible value. It is:

$$R_{opt} = \max \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_g \end{bmatrix} \tag{4}$$

The Euclidean norm of the difference between the predicted weighted rating vector *PR* and the optimal vector R_{opt} can be used as a measure for understanding the service S_{ai}, which is the best for customer C_a. The Euclidean norm will provide the length (size) of the difference between the predicted and the optimal case for each service:

$$diff_{ai} = \left\| R_{opt} - WPR_{ai} \right\| = \left\| \max \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_g \end{bmatrix} - \begin{bmatrix} w_1 * p_{ai1} \\ w_2 * p_{ai2} \\ \vdots \\ w_g * p_{aig} \end{bmatrix} \right\| = \left\| \begin{bmatrix} \max r_1 - w_1 * p_{ai1} \\ \max r_2 - w_2 * p_{ai2} \\ \vdots \\ \max r_g - w_g * p_{aig} \end{bmatrix} \right\| = \left\| \begin{matrix} b_1 \\ b_2 \\ \vdots \\ b_g \end{matrix} \right\| \Rightarrow \tag{5}$$

$$\Rightarrow diff_{ai} = \sqrt{b_1^2 + \cdots + b_g^2}$$

Given this, the smaller $diff_{ai}$ is identifying the service S_{ai} to be selected for customer C_a.

Using this notation and concept, this paper moves on into wrapping up the described model into an algorithm, exposed as a web service in the service lifecycle of the GRIA framework. The effectiveness of this prediction model has been tested in a service provisioning environment using an artificially created dataset. The latter is generated based on an existing smaller dataset using a simple probabilistic model. In that frame and for the purposes of the proposed mechanism evaluation, instead of calculating the whole *PR* vector we focus on predicting the rating value of a single parameter (SLA term). All these are presented in the following Section.

3 Experiment and Evaluation

This section discusses the results to which we concluded by implementing, testing and observing the abovementioned mechanism as a recommender mechanism in the reference architecture. Section 3.1 explains how the recommender system was evaluated for the reference architecture. Section 3.2 presents the way that the evaluation dataset was generated and finally, Section 3.3 presents an evaluation of the results.

3.1 Method of Evaluation

The first step of the evaluation is the creation of an experimental dataset. Given that the reference architecture is not a prevalent market product there was not an adequate number of customers and providers in order to create a full evaluation set. Therefore, the experimental dataset was generated in a way so as to simulate as much as possible a realistic scenario. In this dataset, the ratings of a number of actual business customers were recorded for a number of services and for specific parameters. Further ratings were generated using Mathwave's StatAssist [7] creating a dataset of intuitively adequate size. A part of the dataset was regarded as an evaluation set and it was used so as to measure the deviation of the predicted to the actual values. Moreover,

the recommender system (in the Trusted Third Party side, see Figure 1) was implemented using C++. An Estimator class provides two basic methods:

1. getRatings(): Gathers all the ratings for common user services stored into the TTP Recommender System database.
2. providePredictions(): Predicts customer's ratings for services that he has never invoked in the past, based on related ratings, obtained by the previous method.

After retrieving prediction results for each available service, a Recommender class can compare the services and come to a recommendation for the client (Fig. 2):

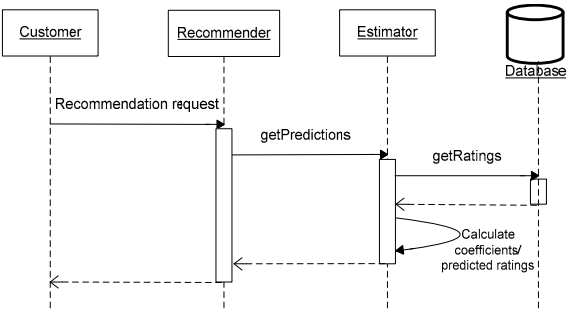


Fig. 2. Recommendation System Components

This implementation was conducted in the frame of the model lifecycle, which is also illustrated in Figure 3.

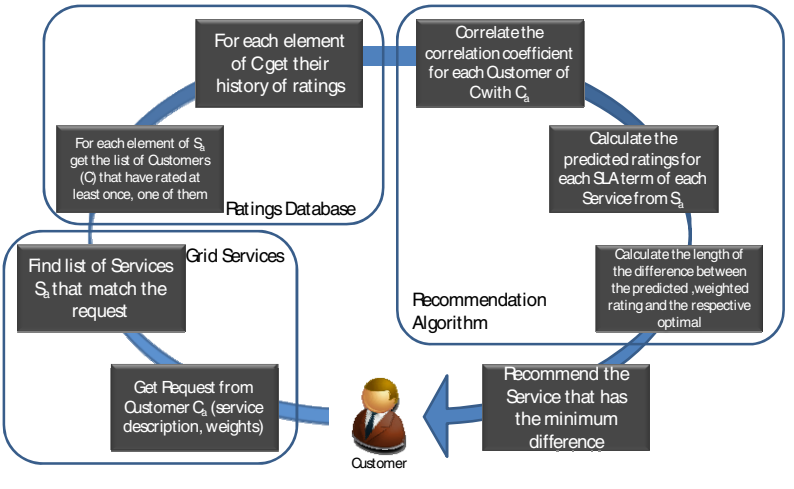


Fig. 3. Recommender System Lifecycle

As depicted in Figure 3, the whole process starts when the customer (say Ca) requests a specific service type. This request is accompanied by the service description (e.g. a service for zipping a file), and the specific requirements expressed through the weights that apply to the SLA terms (e.g. maximum requirements for compression and minimum requirements for delivery time). This is done through the appropriate GRIA interfaces for service selection. The provided requirements are checked against the terms agreed in the SLA for the identification of potential SLA violations and for billing the customer. If there is no SLA violation, the next step in the service invocation lifecycle of GRIA (for more information please refer to [8]) is the service discovery, which includes the discovery of those service types that match the service request. All these take place through a series of interactions between the GRIA Client and Server (see Figure 1).

Following the above processes that are taking place within the Grid infrastructure, the Recommender System, is invoked as a part of a Trusted Third Party's (TTP) operation (Figure 1). The TTP, among others keeps a record of user ratings. Using that record, the Recommender System selects the rating vectors of only those customers that have rated at least one of the available services and at least once.

Then, the mechanism for the recommendation (again belonging to the TTP) is using the produced list of ratings in order to find the correlation coefficient of the Ca ratings with the other customers (that have ratings included in the list). This information is very helpful in order to assign weights to these customers' opinion regarding the service type in question and provide a prediction for each available service instance in this service type. This prediction is produced as a vector of ratings for each service instance. The vector with the smallest difference from the ideal rating of customer Ca also indicates the service instance that should satisfy his needs in the best way. This indication is either delivered to the customer or the GRIA Client is using it to directly invoke the service instance.

For the needs of the experiment, the history records were extended artificially, as described in the following Section.

3.2 Experimental Dataset

The data collected was based on an actual business scenario, that is 3D image Rendering. The scenario involved the provision of various service instances using the GRIA middleware to a number of 3D video animators. The customer (3D animator), invokes a Grid service for transforming the designed wireframe into the desired rendered video frame and in turn to a video stream. The relationship between the animators and the service provider is governed by an SLA, the terms of which are depicted in Table 1. What is of interest in this case is the "QoS Terms" part which represents the application level terms, through which the animator expresses the job requirements.

Table 1. SLA terms for the 3D video rendering service

Generic Terms	QoS Terms	Pricing Terms
Parties	Renderer	Normal Price
Start Time	Resolution	Compensations for all the QoS Terms
End Time	Frame Rate	
	Codec	

For the purposed of the experiment, five different rendering services were set up. The differentiation between them was mainly due to the renderer used (AIR®, 3Delight, AC3D) and the rendering technique (use or no of RenderMan Interface Bytestream (RIB) specification [9]). Seven animators were asked to use the 3D rendering services and to rate -using their experience- each one of the SLA terms according to the provided quality. The rating was on a scale of 1-5 with 1 being poor and 5 being equal to high quality. The animators invoked the services numerous times (from 20 to 30) and the mean value of all ratings was calculated and presented in Table 2.

Table 2. Evaluation of the service quality of 5 services from 7 animators in 4 specific quality parameters

Parameters	Renderer	Resolution	Frame Rate	Codec
Service1	4.3	3.9	3.7	4.1
Service 2	4.5	4.1	3.9	4.3
Service 3	4.4	4.2	3.9	4.3
Service 4	4.4	4.1	4.0	4.3
Service 5	4.4	4.1	4.3	4.4

For the purposes of this model evaluation a great number of customer ratings is required. This was not possible in the frame of the 3D rendering scenario given that only seven animators had used the service for testing purposes. The service was developed for research purposes rather than for commercial which greatly limited the ability to recruit more animators. For that reason there was a need to simulate a greater number of customer ratings using the existing dataset as initiator.

The model that is used below to create an artificial dataset on an available service is based on the Weibull distribution [10]. The Weibull distribution is a continuous probability distribution that is often used:

1. to represent the machine availability in large-scale computing environments such as wide-area enterprises, the Internet and Grids
2. to represent manufacturing and delivery times in industrial engineering
3. in reliability engineering and failure analysis (the most common usage)

The above criteria correspond to some common SLA terms of a service deployed in a service provisioning environment: the availability, the response time and the accuracy of results. Thus, Weibull distribution could also be used to represent the distribution of customers' ratings for simple QoS parameters, such as the service reliability/availability, the time of processing the accuracy of results etc. The Weibull probability density function is:

$$f(x) = \frac{\alpha}{\beta}\left(\frac{x-\gamma}{\beta}\right)^{\alpha-1} e^{-\left(\frac{x-\gamma}{\beta}\right)^{\alpha}}. \quad (6)$$

In this case, $\int_{x1}^{x2} f(x)dx$ represents the probability of a customer's mean rating for a service to fall between values x_1 and x_2.

The distribution parameters are:

α: continuous shape parameter ($a \succ 0$)

β: continuous scale parameter($b \succ 0$)

γ: continuous location parameter ($\gamma = 0$ for the two-parameter Weibull distribution)

Parameter α defines the shape of the distribution, determining how acute is the distribution curve on a mean value, around which most ratings are lying and the variance of the ratings. Hence, when increasing α, the variance of the distribution is reduced and the ratings are gathered around the mean value. Parameter β defines the curve scaling, determining the mean value of the ratings mentioned above, while parameter γ defines the location of the distribution on axis *X*. Since in the current experiment we want the distribution to start at the origin we will consider $\gamma = 0$ in what follows.

By setting different values for the Weibull distribution parameters we can create different distributions with the respective mean and variance. In this case, the decision was to model a large dataset of customer ratings (ranging from 1-5) with Weibull distributions, and also achieve the same mean ratings *m* and variances ($Var(X) = E[(X-m)^2]$) with the ones provided from the small real dataset. To our knowledge, there is no efficient mathematical solution for the problem of estimating Weibull parameters *a* and *b*, while having the distribution mean and variance. The method to create the distributions mentioned above depending on each SLA term mean rating and the ratings variances yields by actual experiments with parameters *a* and *b*, until the desired mean and variance values are approached. Therefore, we used the existing data from the 3D rendering evaluations (Table 2) to set values to the Weibull distribution parameters in order to provide a realistic simulation of the evaluation dataset. In order for the simulation to be valid, the objectivity of the animators' ratings needs to be maintained. Malicious ratings can contribute significantly to the total user rankings, when using the simple median method. However using the proposed method in a long term period, malicious ratings will have a great deviation from most ratings and small relativity across different users, therefore their contribution to a rating prediction for an evaluation parameter will be minimum.

As stated above, there are 5 different Service Providers in the environment. The purpose is to predict the customers' ratings for the 5th Provider, the SLA of which is consisted of 4 parameters: p_1, p_2, p_3 and p_4. So, for each SLA parameter, using the mean values and the variance of the actual ratings in Table 2, 5 Weibull distributions of the users' ratings for the 5 different Providers (20 distributions required) are generated. Some of the mentioned distributions present an almost identical mean value and variance, leading to a reduction of the different Weibull distributions to ten (10). The configuration of these distributions is the following:

Table 3. Configuration of Weibull Distributions for the Various Tested Distributions

Distribution #	a	b	Mean
1	12	4.45	4.3
2	15.5	4.6	4.5
3	16	4.4	4.4
4	16	4.5	4.4
5	10	4.1	3.9
6	11	4.3	4.1
7	11.5	4.4	4.2
8	9.5	3.9	3.7
9	10	4.1	3.9
10	12	4.2	4

In order to provide a visualization of the distribution, the graph for one (the last) configuration is presented:

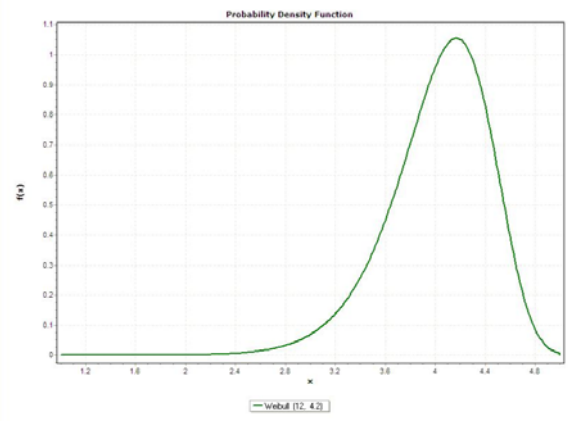


Fig. 4. Weibull Distribution with a=12, b=4.2, Mean Value= 4

The ratings produced for every parameter in this way, also follow a Weibull distribution across different Service Providers. Using these distributions the scale of the problem can be depicted in a better way. The results are presented in the following paragraph.

3.3 Evaluation of Results

The simulation results are presented below. The ratings mentioned above are generated for 500 different users. Each time a prediction for the rating of a user was asked, it was calculated using the Pearson correlation algorithm for all the other values, to correlate the specific user with the other 499 users according to his existing ratings for the other 4 Services. The results presented below occurred by predicting and comparing the actual values with the mean rating for all customers. The blue lines present the realistic values acquired from the Weibull distribution. The red lines present the numbers predicted by the method described above and the purple straight lines present the mean value of the ratings.

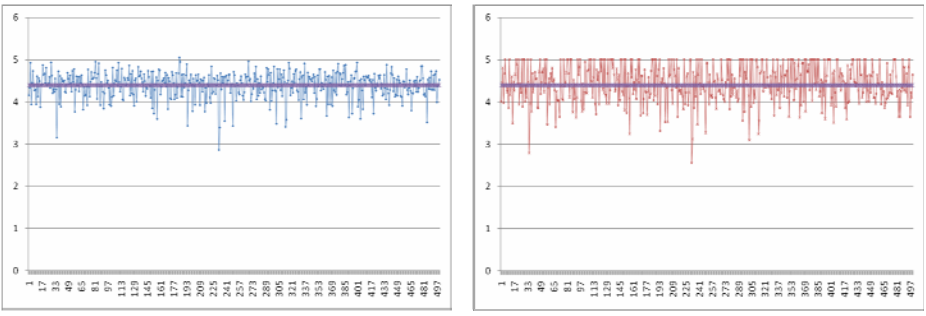


Fig. 5. Real Values and Mean for p_1 (left)/ Prediction Values and Mean for p_1 (right)

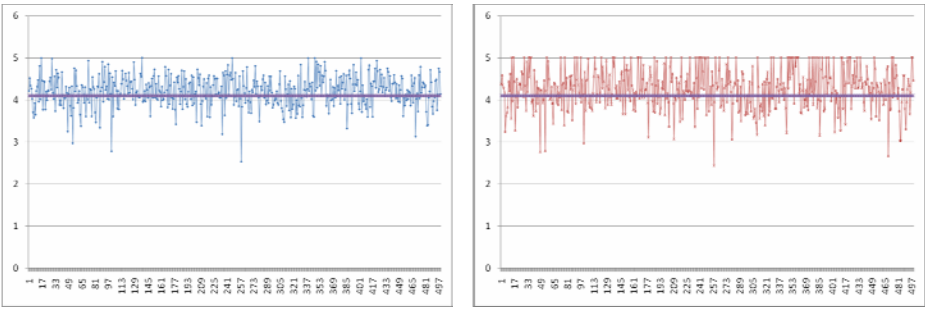


Fig. 6. Real Values and Mean for p_2 (left)/ Prediction Values and Mean for p_2 (right)

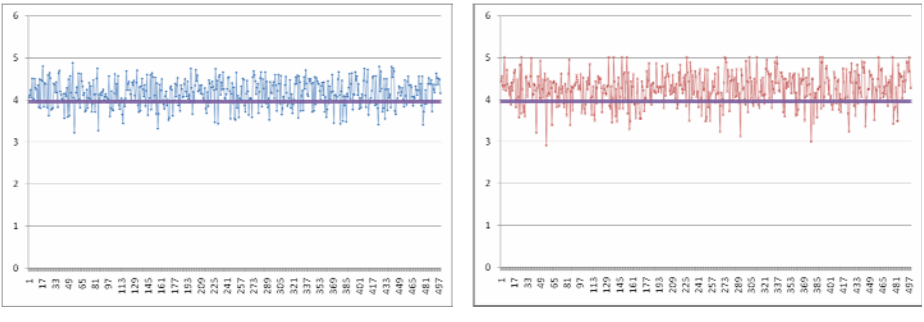


Fig. 7. Real Values and Mean for p_3 (left)/ Prediction Values and Mean for p_3 (right)

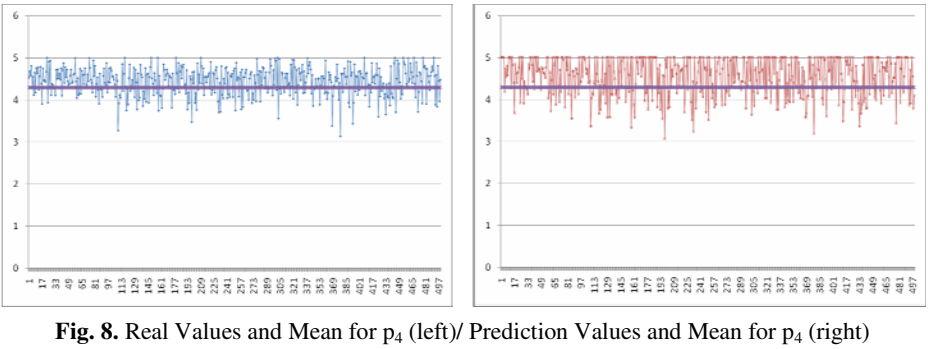


Fig. 8. Real Values and Mean for p_4 (left)/ Prediction Values and Mean for p_4 (right)

As mentioned, in the diagrams above we can see the lines that represent the mean rating of the distribution, the prediction and the actual values of the ratings that were predicted. In most of the cases the predictions are similar to the evaluation set proving the method correct, however, occasionally some divergences from the expected ratings can be spotted. The mean relative error of the above predictions for p_1 is 4.4%. Using the method of mean rating the same error is estimated up to 5.5%. Respectively the same values for p_2 are 4.1% and 7.3%, for p_3 4.3% and 7.3% and for p_4 4.5% and 6.7%. Based on that, the proposed prediction model appears to be better than providing the mean rating in general. The existing divergences can be justified by the fact that the input data are random numbers, generated to follow the Weibull distribution but may deviate occasionally from the selected distribution due to their randomness. By increasing the number of the Service Providers, the mean rating of the users will approach more the mean value of the distribution. Additionally, by increasing the customers in the sample used in this experiment, the predictions are expected to be more accurate, as the contribution of potential malicious ratings will be further reduced and "user groups" are expected to appear, according to customers' preferences, which are going to be really useful to this analysis. By grouping the customers according to their preferences and manipulating these groups, we manage to predict their ratings in a more sufficient and realistic way, as users from inside the same groups are expected to rate in a similar way and not be affected significantly by malicious ratings or data from outside their group.

4 Related Work

Recommender Systems over the internet in general, have been thoroughly studied by various research groups ([11], [12], [13], [14] and others). Collaborative filtering techniques are used to support online Recommender Systems, as can be seen in the works of Herlocker et al [15], Schafer et al [16] and R. Burke [17]. Most of the above also use the Pearson Correlation on mono-dimensional ratings, evaluating user preferences on simple products such as movies, rather than Web Services. Reputation-based Service Selection in SOA has been studied by Ding et al [18], while Grid Service brokering and accounting systems have been presented by and Barmouta and Buyya [19] as well as Venugopal et al [20], not focusing, however, on filtering based on the correlation of user experiences.

This work can be investigated by numerous perspectives making it difficult to find a research studying the exact same topic. Perhaps the most interesting related approach at the same topic is the work of Stefan Schmidt et al [21], where they deal with the topic of decision support for service selection in e-business environments using a fuzzy logic based framework. In this paper, the authors consider selection criteria such as trust, reputation and credibility, which have a fuzzy and variable nature by default [22]. The evaluation of the customer is achieved through the integration of social reputation and trustworthiness factors. The main difference with this work is that the authors are making use of any kind of information that can be translated to the abovementioned selection criteria, rather than using only those that the customer and the provider agree upon. This might have serious implications on the matchmaking process, as there is always a chance to evaluate a service provider in an unfair way or in a way that the customer doesn't agree.

Another interesting work on the field of recommender systems and decision support is [23]. In this paper, the main concept is that the decision making process in e-commerce applications can be assisted by evaluating the past reputation, the trend and the confidence using fuzzy logic techniques. The most interesting part of this work is that it distinguishes the sources from which the three abovementioned criteria are deriving into two categories: The publicly accessible data (such as opinions, ratings, etc) and the private/internal data (past transactions, review results, etc). Using their proposed fuzzy logic mechanism the authors evaluate the consumer value and correlating this information with the consumer risk and business risk, they calculate the consumer value factor. Similarly, the business value factor is calculated and the system attempts to match the two factors so as to optimize the business partnership. Again, the main difference in that the evaluation is not based on criteria that both the negotiating parties agree.

On a different level, an instantiation of the problem that is studied here, is theoretically described in [24]. In the work of Park et al, customer requirements are modeled using utility functions and customers are operating in a non-cooperative multi-class QoS provision model. In this way the authors are emulating the discreet offer model where the provider is offering a service at specific quality levels (bronze, silver, gold service). Thus, the model used for resource sharing and arbitration at a router is used, in the context of QoS, as a way to model delivery of packaged network services by a service provider. Park et al, are studying the activities of the customers in a e-business market as a network game in which the customer policies reach a Nash equilibrium at a specific point of time. The presented model is not possible to be used in such an environment because customers are acting selfishly, therefore, it is not possible to correlate their evaluations and trends. However, this model is somehow more realistic since in a real open market environment, customers are usually forming cliques and trust groups, so as to maximize their cost/quality ratio.

5 Conclusions

In this work we have shown that it is possible to provide decision support services to a customer using other customers' experience that have used the services that are available in a service provisioning environment. This is achieved, by implementing a collaborative filtering mechanism and incorporating the corresponding service in the service lifecycle management of an existing SOA environment. By conducting tests in this environment it was shown that collaborative filtering techniques can help a customer select a service instance that with high probability will fit his requirements in the best way. Also it was shown that it is possible to share information between the customers, by adding an entity playing the role of a trusted third party. The main assumption that this paper makes is that every service provisioning environment will allow for the formation of customer virtual organizations (VO), where the VO manager will act as the respective trusted third party.

The VO manager will only have to collect the opinions of the customers, after invoking a service instance through a rating system. Given that the customers can only evaluate the SLA terms upon which they have agreed for the service invocation (because they proposed them and because they understand them), the system is based

on the rating of each term separately. Therefore, a service instance rating from a customer is basically a vector of ratings rather than a number. By calculating the correlation degree of the customers in the VO using these rating vectors we can end up with recommending a service instance to the customer that is highly correlated to them.

Another approach would be to try to calculate the correlation of the ratings of the customers using only the SLA terms that are in common rather than the common services. The problem of this approach is the different degree of importance that an SLA term has for customers when they are referring to different services. For instance, time, as an SLA term has a different weight when the service type is "Zip" than when it is "Rendering". However, weighting the value of an SLA term according to the service type to which is used makes sense only in the case when the customer has used or is somehow aware of this service type, which usually isn't the case. Therefore, using the SLA terms in service types invoked by both the customers included in the correlation coefficient equation, is the approach that was preferred.

An added value of this method, is that malicious customers that want to manipulate the rating system so as to harm a service provider will be instantly excluded by the decision support model because their opinion will count little to none to the overall prediction. These customers will present big declinations from the norm producing very small values to their correlation coefficients with the customers in question. If this doesn't happen, it will imply that the malicious customer is rating the service in a similar way as the other customers, so still, the result is not affected.

The weakness of the presented model is perhaps when a malicious customer wants to promote or harm a service provider by swarming the system with ratings regarding the provider. However, given the large amount of customers and services that a service provisioning environment is constituted from, the opinion of the individuals counts very little in the vast amount of existing ratings.

Another open issue is the impact of time, since older or newer experiences/ratings by customers should affect differently the overall rating over time. So, in a future work, each rating should normally have a "timestamp" and lose weight, reducing its coefficient, over time. Moreover, low quality experiences in the past might be compensated with good ones recently and the other way around.

Finally, an issue that should be further investigated concerns the assessment of the quality of recommendations by a TTP Broker. The matter of trust for filtering and weighting recommendations is always existent in the Internet of Services and it has been studied in several works, (i.e. Ries and Heinemann [25]).

In any case, building and maintaining the customer's trusted network could probably complete this work. This could be done by either using Bayesian networks (calculating the probability of trusting someone given that it hasn't be proved wrong in the past) or by using the social graph. The latter provides a very interesting perspective to the concept upon which this paper is structured, given that Web 2.0 and especially social networking applications seem to be governing technologies in the future internet. However, both these ideas remain to be implemented and evaluated in future work.

References

1. Masclet, D., Pénard, T.: Is the eBay Feedback System Really Efficient? An Experimental Study, Working Paper, University of Caen (2008)
2. Pennock, D., Horvitz, E., Lawrence, S., Giles, L.: Collaborative filtering by personality diagnosis: A hybrid memory- and model-based approach. In: Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence, pp. 473–480. Morgan Kaufmann, San Francisco (2000)
3. Breese, J., Heckerman, D., Kadie, C.: Empirical Analysis of Predictive Algorithms for Collaborative Filtering. In: Proceedings of the14th Annual Conference on Uncertainty in Artificial Intelligence, Madison, WI, pp. 43–52 (1998)
4. Snelling, D., Anjomshoaa, A.: NextGRID Architectural Concepts. In: CoreGRID Symposium 2007, Rennes, France, August 27-28 (2007)
5. Surridge, M., Taylor, S., De Roure, D., Zaluska, E.: Experiences with GRIA-Industrial Applications on a Web Services Grid. In: Proceedings of the First International Conference on e-Science and Grid Computing, pp. 98–105. IEEE Press, Los Alamitos (2005)
6. GRIA, Service Oriented Collaborations for Industry and Commerce, `http://www.gria.org`
7. Mathwave StatAssist, `http://www.mathwave.com/products/assist.html`
8. Kyriazis, D., Tserpes, K., Menychtas, A., Sarantidis, I., Varvarigou, T.: Service Selection and Workflow Mapping for Grids: An approach exploiting Quality of Service Information. Concurrency and Computation: Practice and Experience 21(6), 739–766 (2008)
9. RenderMan Interface Specification v3.2, `http://tinyurl.com/RIBSpec`
10. Weibull, W.: A statistical distribution function of wide applicability. J. Appl. Mech. 18, 293–297 (1951)
11. Resnick, P., Varian, H.: Recommender systems. Communications of the ACM 40(3), 56–58 (1997)
12. Schafer, B., Konstan, J., Riedl, J.: Recommender systems in e-commerce. In: Proceedings of the 1st ACM conference on Electronic commerce, Colorado USA, pp. 158–166 (1999)
13. Ramakrishnan, N., Keller, R., Mirza, B., Grama, A., Karypis, J.: Privacy risks in recommender systems. IEEE Internet Computing 5(6), 54–62 (2001)
14. Ansari, A., Essegaier, S., Kohli, R.: Internet Recommendation Systems. Journal of Marketing Research 37(3), 363–375 (2000)
15. Herlocker, J., Konstan, J., Teerven, L., Riedl, J.: Evaluating Collaborative Filtering Recommender Systems. ACM Transactions on Information Systems (TOIS) 22(1), 5–53 (2004)
16. Schafer, B., Frankowski, D., Herlocker, J., Sen, S.: Collaborative filtering recommender systems. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.) Adaptive Web 2007. LNCS, vol. 4321, pp. 291–324. Springer, Heidelberg (2007)
17. Burke, R.: Integrating knowledge-based and collaborative-filtering recommender systems. In: Proceedings of the Workshop on AI and Electronic in Commerce, pp. 69–72 (1999)
18. Ding, Q., Li, X., Zhou, H.: Reputation based Service Selection in Grid Environment. In: Proceedings of the 2008 International Conference on Computer Science and Software Engineering, pp. 58–61 (2008)
19. Barmouta, A., Buyya, R.: GridBank: A Grid Accounting Services Architecture (GASA) for Distributed Systems Sharing and Integration. In: Proceedings of the 17th International Symposium on Parallel and Distributed Processing, p. 245 (2003)

20. Venugopal, S., Buyya, R., Winton, L.: A Grid Service Broker for Scheduling Distributed Data-Oriented Applications on Global Grids. In: Proceedings of the 2nd workshop on Middleware for grid computing, Ontario, Canada, pp. 75–80 (2004)
21. Schmidt, S., Steele, R., Dillon, T., Chang, E.: Fuzzy Decision Support for Service Selection in E-Business Environments. In: Proceedings of IEEE Symposium on Computational Intelligence in Multicriteria Decision Making, Honolulu, USA, pp. 374–381 (2007)
22. Chang, E., Dillon, T.: Fuzzy nature of trust and dynamic trust modeling in service oriented environments. In: Proceedings of the 2005 Workshop on Secure Web Services, pp. 75–83 (2005)
23. Gu, X., Zhu, Q.: Fuzzy multi-attribute decision-making method based on eigenvector of fuzzy attribute evaluation space. Decision Support Systems, Elsevier 41, 400–410 (2006)
24. Park, K., Sitharam, M., Chen, S.: Quality of service provision in noncooperative networks with diverse user requirements, Special issue on information and computational economics. Decision Support Systems, 101–122 (2000)
25. Ries, S., Heinemann, A.: Analyzing the robustness of CertainTrust. In: International Federation for Information Processing (IFIP), vol. 263, pp. 51–67 (2008)

Resource-Level QoS Metric for CPU-Based Guarantees in Cloud Providers

Íñigo Goiri, Ferran Julià, J. Oriol Fitó, Mario Macías, and Jordi Guitart

Barcelona Supercomputing Center and Universitat Politecnica de Catalunya
Jordi Girona 31, 08034 Barcelona, Spain
{igoiri,fjulia,fito,mario,jguitart}@ac.upc.edu

Abstract. Success of Cloud computing requires that both customers and providers can be confident that signed Service Level Agreements (SLA) are supporting their respective business activities to their best extent. Currently used SLAs fail in providing such confidence, especially when providers outsource resources to other providers. These resource providers typically support very simple metrics, or metrics that hinder an efficient exploitation of their resources.

In this paper, we propose a resource-level metric for specifying fine-grain guarantees on CPU performance. This metric allows resource providers to allocate dynamically their resources among the running services depending on their demand. This is accomplished by incorporating the customer's CPU usage in the metric definition, but avoiding fake SLA violations when the customer's task does not use all its allocated resources. As demonstrated in our evaluation, which has been conducted in a virtualized provider where we have implemented the needed infrastructure for using our metric, our solution presents fewer SLA violations than other CPU-related metrics.

Keywords: QoS metrics, SLA, Cloud provider.

1 Introduction

The emergence of Cloud computing solutions has attracted many potential consumers, such as enterprises, looking for a way to reduce the costs associated with supporting their business processes. Using the Cloud, these customers can outsource services (offered by service providers) which can be easily composed to build distributed applications. Additionally, the Cloud allows resource providers to offer raw resources as a service where the consumers can outsource the execution of their own services. In the same way, service providers can also use the Cloud to overcome an unexpected demand on their services by outsourcing additional resources to other providers, or even they can fully rely on external resources to provide their services. In all these scenarios, it is highly desirable that the consumers receive fine-grain Quality of Service (QoS) guarantees from the providers. However, this becomes essential when service providers outsource

J. Altmann and O.F. Rana (Eds.): GECON 2010, LNCS 6296, pp. 34–47, 2010.

resources to resource providers, since the QoS guarantees they can provide to their respective customers depend on the QoS guarantees they receive from the resource providers.

Typically, a provider agrees the QoS with its customers through a Service Level Agreement (SLA), which is a bilateral contract between the customer and the provider (or between providers) that states not only the conditions of service, but also characterizes the agreed QoS between them using a set of metrics. Service providers naturally offer service-level metrics (e.g. service execution deadline [6]) to their customers for specifying the QoS. Using service-level metrics has advantages for both the customer and the provider. The former does not need to provide detailed information about the resource requirements of the service to be executed (probably the customer does not know this exactly), but only a high-level performance goal (e.g. a deadline for executing the service). The latter can freely decide the allocated resources to the service whereas it guarantees that the service meets the agreed performance goals. Being able to dynamically allocate resources to the different services is especially important for Cloud providers considering that, aiming for profitability, they tend to share their resources among multiple concurrent services owned by different customers.

On the other hand, resource providers in the Cloud must offer resource-level metrics that can be used to provide fine-grain QoS guarantees. First, the QoS agreement can be naturally expressed using resource-level metrics (e.g. number of processors, frequency of processors, FLOPS, etc.), since raw resources are the traded good. Second, having fine-grain metrics, which guarantee a given resource allocation during a time period, is especially important for service providers that outsource resources to resource providers, as we have stated before. For instance, try to figure out how a service provider could guarantee that a service will finish within a given deadline if he does not receive fine-grain guarantees on the FLOPS supplied at every instant by the resource provider where the service is running.

Furthermore, some service providers can also benefit in some situations from supporting resource-level metrics. First, those that do not support the inference of the resource requirements of a service to fulfill a given service-level metric, and for this reason, cannot offer service-level metrics in the SLA. Second, those dealing with customers (typically coming from the HPC domain) that prefer to stay with the resource-level metrics that they have been using for a long time, instead of moving to service-level metrics.

Nevertheless, current Cloud providers do not support fine-grain resource-level QoS guarantees on their SLAs. In fact, most of them only support SLAs with very simple metrics based on resource availability [2,12,14]. For instance, whereas Amazon EC2 [4] offers different instances according to their computing capacity (which is measured in ECUs, EC2 compute units), there is not any guarantee in the SLA that this computing capacity will be supplied during the whole execution of the instance, as Amazon's SLAs only provide availability guarantees [12].

According to this, one could think on using Amazon's ECUs to support fine-grain resource-level guarantees on Cloud SLAs, considering also the porting of traditional resource-level metrics from the Grid environment to Cloud providers.

However, this must be carried out carefully, since it can prevent the providers from obtaining the maximum profit of their resources if done naively. In particular, metrics related to the provider's computing capacity (i.e. CPU) are especially susceptible to naive usage. For instance, if the agreed metric in the SLA establishes that the number of processors allocated to a given service must be greater or equal to some value, the provider must maintain this assignment during the whole execution of the service, even if that service is not using all the allocated capacity during some phases of its execution (i.e. the provider is forced to statically overprovision processors to the service to avoid SLA violations). Notice that the unused resources could be temporarily allocated to another service, improving in this way the utilization and the profit for the provider.

In this paper, we derive a resource-level metric for specifying fine-grain QoS guarantees regarding the computing capacity of a Cloud provider by extending the Amazon's approach. Our metric overcomes the limitations of traditional CPU-related metrics. By taking into account the customer's resource usage, it allows the provider to implement dynamic resource provisioning and allocate to the different services only the amount of CPU they need. In this way, better resource utilization in the provider can be achieved. In addition, it avoids fake SLA violations, which we define as those situations where the SLA evaluator mechanism detects that an SLA is being violated, the provider will be penalized for this, but the violation is not provoked by the provider's resource allocation. In general, this occurs when the provider uses a poorly designed resource-level metric and the customer's service does not use all the resources it has allocated. Of course, the customer is free to use the amount of resources he wants, and this must not provoke any SLA violation. Therefore, the solution is to design solid resource-level metrics that support this. Finally, the proposed metric can be used in heterogeneous environments, since it is based on Amazon's ECUs.

The remainder of this paper is organized as follows: Section 2 explains our approach to deal with platform heterogeneity. Section 3 describes the derivation of our resource-level CPU metric. Section 4 presents the experimental environment where this resource-level metric has been implemented. Section 5 describes the evaluation results. Section 6 presents the related work. Finally, Section 7 presents the conclusions of the paper and the future work.

2 Metric Unification among Heterogeneous Machines

Cloud providers typically present very diverse architectures: different processor models, each of them with different clock frequencies, etc. For this reason, a good resource-level metric has to be platform-independent so it can be used in all these architectures. In this section, we describe our approach for unifying computing capacity metrics among machines with heterogeneous architectures.

Commonly, *megahertz* have been used to measure the computing capacity of a machine. Nevertheless, this does not directly measure the computing power of a machine, since noticeable differences can be observed depending on the processor architecture. For instance, using this measure a Intel Pentium III with 500 MHz

would be 20 times slower than a Intel Xeon 4-core with 2.6 GHz ($\frac{4 \cdot 2600}{500} = 20$). However, simple tests demonstrate that it can be up to 85 times slower.

In order to consider the heterogeneity of the different processor architectures, Amazon uses EC2 compute units (ECU) in its services [4]. An ECU is equivalent in CPU power to a 1.0-1.2 GHz 2007-era AMD Opteron or Intel Xeon processor. This serves as a unified measure for the computing power, though it is not easily portable among different architectures. In this work, we use ECUs in order to unify CPU-related SLA metrics among heterogeneous machines, and additionally we extend the Amazon's approach in order to set up SLAs that provide fine-grain CPU guarantees based on ECUs during a time period.

In our proposal, the maximum computing capacity of a given machine is measured using its maximum amount of CPU[1] ($maxCPU$) and the $ECUs$ associated to the processor installed in that machine ($\frac{maxCPU}{100} \cdot ECUs$).

3 Derivation of CPU-Based SLA Metric

This section describes how our resource-level metric for establishing computing power guarantees is derived, and at the same time, discusses the limitations of alternative metrics. All the metrics discussed in this section intent to establish a guarantee on the computing performance (in terms of CPU) of a service over a time period in a given provider using the idea presented in the previous section. This guarantee is a fixed value for each SLA, represented by the SLA_i term in the formulas, which results from the negotiation between the customer and the provider. The customer is only required to specify the computing performance he requires (in terms of ECUs), which will be accepted by the provider if he is able to provide that performance.

The main difference among the metrics is how the amount of CPU for a service is defined. The more natural approach is specifying CPU performance as a function of the allocated CPU to a service, as shown in Equation 1. This metric specifies that the computing power for a service i at every time period t has to be at least the agreed value in the SLA (SLA_i) and depends on the amount of CPU that the provider assigns to the service in that time period ($assig_i(t)$). This assignment can vary over time depending on the provider's status, and it is periodically obtained by means of the monitoring subsystem.

$$\frac{assig_i(t)}{100} \cdot ECUs \geq SLA_i \tag{1}$$

Note that using this metric forces the provider to statically allocate to each customer at least the amount of CPU agreed in the SLA (he can assign more if he wants), because otherwise, the SLA will be violated. Note that there is not any control on whether the customer uses its allocated CPU or not. This is a quite restrictive approach, especially when the customers' services have a variable

[1] All CPU-related measures are quantified using the typical Linux CPU usage metric (i.e. for a computer with 4 CPUs, the maximum amount of CPU will be 400%).

CPU usage over time. In this case, the provider will suffer from low resource utilization when the services do not use all the CPU they have allocated, since unused resources cannot be allocated to other services running in the provider.

As we have commented before, the provider aims to dynamically allocate its resources to the services depending on their demand, in order to improve resource utilization (and then increase profit). This requires an SLA metric that considers the CPU usage of the services. However, as the CPU usage depends on the client's task behavior, it must be carefully used as a CPU guarantee because it can provoke undesired effects.

For instance, Equation 2 shows an SLA metric where the computing power for a service i at every time period t depends on the amount of CPU that the service uses in that time period ($used_i(t)$). This metric assumes a provider that is able to predict the CPU usage of a given service during the next time period using the CPU usage measures of the service from the previous time periods. This could be achieved with reasonable accuracy using techniques such as Exponential Weighted Moving Average (EWMA). The provider will assign CPU to the service according to its CPU usage prediction. This allows the provider to dynamically allocate the CPU among the services whereas it assigns each service at least with the CPU required to fulfill the SLA.

$$\frac{used_i(t)}{100} \cdot ECUs \geq SLA_i \tag{2}$$

When using this metric, an SLA could be violated in two situations. First, when the provider assigns to the service an amount of CPU that is not enough to fulfill the SLA. This is a *real* violation, the provider is responsible for it, and must pay the corresponding penalty. Second, when the provider assigns to the service an amount of CPU that should be enough to fulfill the SLA, but the service does not use all the assigned CPU. This is what we have defined as *fake* violation, since the provider is not causing it, and for this reason, he should not pay any penalty. However, Equation 2, as currently defined, provokes this to be considered as a *real* violation, thus penalizing the provider for it.

Of course, the service should be able to use only a part of the CPU it has assigned without incurring on SLA violations. In order to allow this, we introduce our metric, in which we introduce a factor that represents the percentage of CPU used by the service with respect to its total amount of allocated CPU. As shown in Equation 3, when the service is using all the assigned resources, Equation 1 applies, so an SLA violation will only arise when the assigned resources are not enough to fulfill the SLA. When the service is not using all the allocated resources then the SLA is considered to be fulfilled, since the provider is not responsible that the service does not exploit its allocated resources.

$$\frac{assig_i(t)}{100} \cdot ECUs \geq SLA_i \cdot \left\lfloor \frac{used_i(t)}{assig_i(t)} \right\rfloor \tag{3}$$

However, some services, even being CPU-intensive, do not use the 100% of their assigned CPU during their whole execution. For these services, Equation 3 does

not work. In order to overcome this limitation, we introduce an α factor as shown in Equation 4. This factor acts as a threshold to choose when the service is considered to be using all its assigned resources. For instance, if we consider that a service is using all its allocated resources when it reaches a 90% utilization, α should be set to 0.1.

$$\frac{assig_i(t)}{100} \cdot ECUs \geq SLA_i \cdot \left\lfloor \frac{used_i(t)}{assig_i(t)} + \alpha \right\rfloor \tag{4}$$

Operating on Equation 4, we obtain the version of our metric ready to be used in an SLA:

$$\frac{\frac{assig_i(t)}{100} \cdot ECUs}{\left\lfloor \frac{used_i(t)}{assig_i(t)} + \alpha \right\rfloor} \geq SLA_i \tag{5}$$

Notice that equation in this form can have an undefined value when the denominator is zero, which happens when the service is not considered to use all its allocated resources. We avoid this by defining SLA'_i as $1/SLA_i$ and operating the equation. Notice that, when using this metric, the value specified in the SLA is SLA'_i instead of SLA_i. The final version of our metric is as follows:

$$\frac{\left\lfloor \frac{used_i(t)}{assig_i(t)} + \alpha \right\rfloor}{\frac{assig_i(t)}{100} \cdot ECUs} \leq \frac{1}{SLA_i} = SLA'_i \tag{6}$$

4 Experimental Environment

In order to compare how our metric performs with respect to other CPU-based metrics, we use the SLA enforcement framework built on top of the virtualized provider presented in [15]. This provider, which has been developed within the BREIN European IST Project [9], uses virtual machines to execute the tasks, which allows taking advantage of virtualization features such as migration and easy resource management. In addition, virtualization allows the consolidation of services in the same physical resources, which enhances resource utilization. However, if the amount of resources assigned to these services is static and does not consider the real resource usage, the underutilization problem remains, as we have discussed in this paper. The virtualized provider also has a monitoring subsystem that allows easily consulting the amount of CPU allocated to a VM ($assig_i$) and the amount that it is really using ($used_i$). In addition, the ECUs of the underlying machine can be also calculated from the processor model, its frequency, and its associated *BogoMips* [11]. These *BogoMips* are used to convert the computing power among different architectures.

The SLA framework allows assigning its own SLA to each service by using a XML description that combines both WS-Agreement [5] and WSLA [16] specifications. Using these specifications, we can accurately define the metrics

derived in the previous sections. In addition, we use some features of the above-mentioned SLA specifications to define the window size of the average (i.e. 10) and the interval between two consecutive measures (i.e. 2 seconds).

Each SLA S_i specifies the revenue that the customer will pay if the SLA is fulfilled ($Rev(S_i)$), and the penalty that the provider will pay otherwise ($Pen(S_i)$). According to this, the provider's profit for running a given application ($Prof(S_i)$) results from the revenue paid by the customer minus the penalties that the provider has to pay due to SLA violations, i.e. $Prof(S_i) = Rev(S_i) - Pen(S_i)$.

In order to establish the penalties, we use a methodology similar to the one presented in [13], which is built over the same Cloud infrastructure. Each penalty $Pen(S_i)$ is calculated as a percentage of the revenue obtained when fulfilling the corresponding SLA in the following way: $Pen(S_i) = Rev(S_i) \cdot \frac{Gom(\sigma(S_i))}{100}$. This percentage is calculated by using a Gompertz function, which is a kind of *sigmoid function*. Its basic form is $y(t) = ae^{be^{ct}}$, where a is the upper asymptote, c is the growth rate, and b, c are negative numbers.

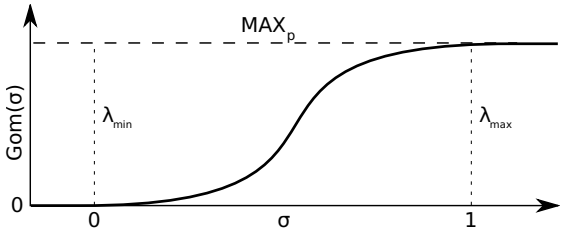


Fig. 1. Gompertz Function for SLA Penalties

For our purposes, we have adapted this function as shown in Figure 1, which displays the penalty percentage depending on a $\sigma(S_i)$ function that represents the SLA violation ratio. In particular, as shown in Equation 7, when this ratio is zero or less, the penalty percentage is 0. When this ratio is one, the percentage tends to $MAX_P\%$ of the price that the client pays for SLA S_i. Notice that this upper limit ($MAX_P\%$ of the price) can be agreed with the client during the SLA negotiation process.

$$Gom(\sigma(S_i)) = \begin{cases} 0 & \text{if } \sigma(S_i) \leq 0 \\ MAX_P \cdot e^{-e^{-7 \cdot \sigma(S_i) + e^1}} & \text{otherwise} \end{cases} \quad (7)$$

As shown in Equation 8, $\sigma(S_i)$ function depends on the number of violations occurred for that SLA (V_i) and is parameterized with two thresholds, λ_{min} and λ_{max}, which indicate the minimum number of SLA violations in order to start paying penalties and the maximum number of SLA violations that could occur during the execution of the application, respectively.

$$\sigma(S_i) = \frac{V_i}{\lambda_{max} - \lambda_{min}} - \frac{\lambda_{min}}{\lambda_{max} - \lambda_{min}} \quad (8)$$

5 Evaluation

In this section, we compare how our metric performs with respect to the other CPU-related metrics introduced in Section 3. First, we use a small proof-of-concept workload composed of three tasks to show in detail the consequences of using each metric and then, we use a real workload to evaluate each metric.

5.1 Proof-of-Concept Workload

We have conducted three experiments. Each of them consists of the concurrent execution of three tasks, which are basically CPU-consuming tasks with variable CPU requirements over time, in a single node of the virtualized provider during 400 seconds. This makes the system able to dynamically change the resources allocated to each virtual machine and allows demonstrating the benefits of our metric in such a scenario. This node is a 64-bit architecture with 4 Intel Xeon CPUs at 2.6GHz and 16GB of RAM, which runs Xen 3.3.1 [22] in the Domain-0. This processor has been measured to have 5322.20 BogoMips, which corresponds to 10.4 ECUs approximately. Each task is started at a different time to simulate what could happen in a real scenario, where different customers can reach the provider at different times.

Each task has its own SLA, which describes the agreed QoS with the provider. For each experiment, a different SLA metric is used to specify this QoS. In particular, in the first experiment all tasks use the metric referred in Equation 1 (from now on denoted as *SLA Metric A*), in the second one they use the metric referred in Equation 2 (denoted as *SLA Metric B*), and in the last one they use the metric referred in Equation 6 (denoted as *SLA Metric C*).

For each experiment, we have monitored the allocated CPU to each task, its real CPU usage, and the value of the SLA metric at each moment in time. Comparing this value with the guarantee agreed in the SLA, we can know if the SLA is being violated or not. Figure 2 displays all this information. The top graphic in this figure shows the CPU assignment and usage for each task over time. These values will be the same independently of the metric used in the SLA. We have forced the provider to execute the three tasks, but there are not enough resources for fulfilling all the SLAs. This allows us evaluating how the different metrics deal with real SLA violations. The provider distributes the resources in such a way that the SLAs of *Task2* and *Task3* are never violated in these experiments, while *Task1* behavior shows all the possible situations regarding SLA management. For this reason, we focus the explanation only on *Task1*, which has negotiated an SLA with a CPU requirement of 6 ECUs (1/6 for Metric C).

As shown in the top graphic of Figure 2, at the beginning of the experiment *Task1* is assigned with the whole machine, since it is the only task running in the provider. When *Task2* arrives at the provider at second 150, the CPU is redistributed among *Task1* and *Task2* according to their requirements. Finally, at second 250, *Task3* arrives at the provider, which redistributes again the CPU among the three tasks. Regarding the CPU usage of *Task1*, it starts its execution

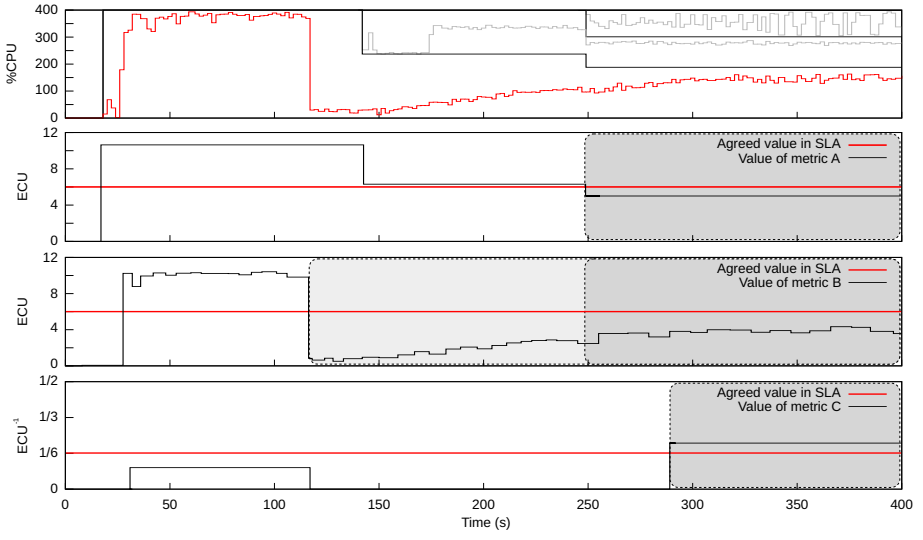


Fig. 2. CPU-Based SLA Metrics Comparison

consuming almost 400% of CPU. After 100 seconds of execution, it reduces its CPU consumption and starts using just 20% of CPU for the next 40 seconds. Then, it begins increasing its CPU usage until it consumes around 150% of CPU.

Next three graphics in Figure 2 show the value of the SLA metric over time and its agreed value in the SLA, for Metric A, B and C, respectively. As described in Section 3, *SLA Metric A* only takes into account the amount of resources assigned to the task. While this value is above of the agreed value in the SLA, SLA is fulfilled. Otherwise, an SLA violation arises. As shown in the second graphic, due to the redistribution of CPU occurred when *Task3* arrives at the provider at second 250, the value of the *SLA Metric A* for *Task1* falls below the agreed value in the SLA, thus violating it. The shaded zone identifies in the figure the interval where *Task1* is violating the SLA (from second 250 to 400). In order to avoid this, the provider would be forced to refuse executing *Task3*, even if this would be initially possible since the other tasks are not using all their allocated CPU. Notice that using this metric makes the provider unable to exploit adequately its resources, in the sense that it should always assign enough resources to fulfill the SLA, despite that tasks use them or not.

On the other side, *SLA Metric B* considers the resource usage of the tasks. Using this metric allows the provider moving freely the CPU assignment if tasks do not use all their allocated resources. Nevertheless, as shown in the third graphic, this metric can give fake SLA violations. In particular, when the CPU usage of *Task1* goes below the agreed value in the SLA (from second 115 to 400), the SLA is being violated. However, not all these violations are a responsibility of the provider. In particular, between second 115 and 250 (light gray zone in the graphic), the SLA is being violated because although the provider is giving

enough resources to guarantee the SLA, the task does not use all its assigned CPU. As we have described before, this is a fake violation. On the other side, from second 250 to 400, the SLA is being violated (dark gray zone in the graphic) because the provider has not allocated enough resources to the task to fulfill its SLA, and consequently, even if it used all of them, its CPU usage would be under the SLA threshold. This is then a real violation. Notice that the value of *SLA Metric B* has some start delay with respect to the CPU usage. This is because of the different sampling rates between the SLA monitoring system and the CPU usage monitoring system.

Finally, *SLA Metric C* (the one proposed in this paper) gets the best from the two other ones, since it considers both the CPU assigned to a task and its real CPU usage. As discussed in Section 3, this metric specifies the inverse value in the SLA with respect to the other metrics. For this reason, when using this metric, an SLA violation will happen when the measured value of the metric is greater than the agreed one (1/6 in this particular experiment). As shown in the fourth graphic, this only occurs in the shaded zone, that is, from second 290 to 400. As commented before, this is the only zone where the task is using all its allocated resources and the provider has not allocated enough resources to the task to fulfill its SLA. On the other side, notice that when using this metric fake SLA violations are avoided, even when the task is not using all its allocated resources (as occurs from second 115 to 290). In these experiments, we have considered that a task is using all its allocated resources when it uses more than 70% of them (this corresponds to a 0.3 value for the α factor in Equation 6).

Summarizing, three well-differentiated phases can be appreciated in the figure. In the first one, the assigned CPU is enough to fulfill the SLA and *Task1* is using all its allocated resources. Consequently, none of the metrics is violating the SLA. In the second one, CPU allocated to *Task1* has been reduced, but the task is not using all its allocated resources. In this case, using the *SLA Metric A* causes low resource utilization because unused resources cannot be assigned to any other task without violating the SLA, while using the *SLA Metric B* causes fake SLA violations. Finally, in the last phase, the resources assignment is not enough to fulfill the SLA, and *Task1* uses all its assigned resources. In this case, all the metrics violate the SLA.

5.2 Real Workload

In this section, we evaluate the different metrics by executing a real workload. The workload is a whole day load obtained from Grid5000 [1] corresponding to Monday first of October of 2007. 215 jobs of 10 different applications are submitted during this period. The size of our real testbed is not enough to execute this workload. For this reason, we have built a simulated provider composed by 24 nodes, which mimics the behavior of the virtualized provider used in previous experiments. The number of nodes has been chosen in order to able to execute the maximum number of concurrent jobs of the workload, while still violating some SLAs to demonstrate the behavior of the different metrics.

We set up three types of nodes in the simulated provider in order to demonstrate how our proposal can deal with heterogeneous machines. The first type is the one already used in previous experiments. The other two nodes types comprise a 64-bit architecture with 4 Intel Xeon CPUs at 3GHz and 16GB of RAM, and a 64-bit architecture with 8 Intel Xeon CPUs at 2.8GHz and 16GB of RAM, which respectively correspond to 12 and 22.4 ECUs. For each task, we specify a static requirement of resources, and the dynamic resource usage it performs during its execution. Each task is executed within a VM with 2 VCPUs, since this is enough to fulfill the CPU requirements of all the applications. In order to demonstrate the impact on each metric of having dynamic CPU usage pattern over time, we have forced two of the 10 applications to use more CPU than the requested during some parts of their execution. Resource requirements and usages constitute the input of the simulator, which implements a backfilling policy taking into account these requirements in order to decide the allocated resources to each task and simulates the specified resource consumption for each task in a given time slot.

Each task has a different SLA. All of them use the same metric for specifying the CPU guarantee, but with a different agreed value, which depends on the task CPU requirement. In addition, the SLA specifies a revenue of $0.38 per hour (we use the pricing of an Amazon EC2 medium instance [4], which has similar features to the VMs used in the experiment). The SLA is evaluated every 100 seconds, and the penalty is calculated following Equation 7 using a maximum penalty (MAX_P) of 125%. As the SLA is evaluated every 100 seconds, the maximum number of SLA violations (λ_{max}) occurs when the SLA is violated every sample. Hence, λ_{max} is equal to the number of samples, which is calculated by dividing the application execution time (T_{exec}) by the sampling period (T_{sample}). Using this information, and considering that λ_{min} is zero in our case, we can operate on Equation 8 and calculate $\sigma(S_i)$ as follows: $\sigma(S_i) = \frac{V_i \cdot T_{sample}}{T_{exec}}$.

Table 1 shows the obtained results using each one of the metrics, including the number of violations (real, fake, and total) and the final profit. For *SLA Metric C*, we have also evaluated the impact of the α factor. These results demonstrate that using our metric the provider can get the highest profit, since it avoids all the fake SLA violations and reduces the real ones when the application is not using all the allocated resources. In particular, we have reduced the incurred penalties in more than a 58% regarding *SLA Metric A* and more than a 72% compared to *SLA Metric B* when $\alpha = 0.1$. In addition, note that the lower the alpha factor is, the lower the number of SLA violations is, though the difference is not very noticeable. Nevertheless, this difference highly depends on the CPU usage of the applications. This occurs because for higher alpha values the task is considered to be using all its allocated resources during more time, which increases the probability of SLA violations imputable to the provider.

These results allow us to conclude that our metric has globally less SLA violations than the other ones, it is able to avoid fake SLA violations, and it allows the provider to freely redistribute the resources when the tasks are not using them.

Table 1. Number of SLA Violations and Profit (Real Workload)

SLA Metric (α)		Real	Fake	Total	Profit ($)
A		3891	0	3891	231.0
B		3891	2857	6748	176.6
C	0.05	1674	0	1674	277.5
C	0.1	1782	0	1782	271.3
C	0.2	1877	0	1877	266.9
C	0.4	1884	0	1884	266.6

6 Related Work

As Cloud technology is still in its early stage, current Cloud providers only support the specification of QoS guarantees by means of SLAs in a very simple way. For instance, the SLAs supported by Amazon EC2 [4] only consider an availability metric, so-called *annual uptime percentage* [12]. It is noticeable that the violations must be monitored and claimed by the user. Furthermore, whereas Amazon EC2 offers different instances according to their ECUs, there is not any guarantee in the SLA that this computing capacity will be supplied during the whole execution of the instance.

Similarly, GoGrid also considers availability metric [14], including the availability of the storage and the primary DNS. It also offers some metrics related with network performance such as jitter, latency, and packet loss rate. Analogously, 3Tera also provides availability guarantees on its Virtual Private Datacenter [2]. Nevertheless, none of the Cloud approaches for SLA considers CPU-related metrics since they implement static provisioning of resources. As this can lead to low resource utilization, moving to dynamic provisioning is recommended to increase profit.

Until now, SLAs have been primarily used to provide QoS guarantees on e-business computing utilities. Due to this, most of the works focusing in this area [3,7,8,17,23] have used metrics of interest in these environments such as throughput, response time, and availability. None of them has included resource-level metrics as part of its solution, as we suggest in this paper.

On the other side, some effort has been carried out in the Grid environment in order to provide QoS guarantees using SLAs. Some proposals focus on service-level metrics. For instance, [10,20] propose SLA management systems for Grid environments which use the expected completion time of the jobs for defining SLA metrics. Similarly, [18] defines latency, throughput, and availability metrics and tries to minimize execution times and costs in a Grid provider. Whereas these works neglect resource-level metrics in their solution, other works define metrics for providing CPU guarantees in the SLA, but usually in a very limiting way for the resource provider. For instance, [21] allows defining the CPU count, the CPU architecture, and the CPU performance when executing a physics application on the Grid. Similarly, [19] uses metrics such as the number of processors, the CPU share, and the amount of memory. As we have discussed, when used incorrectly, some these metrics force the provider to statically overprovision CPU to the

applications in order to avoid SLA violations, while others induce fake SLA violations when the applications do not use all their allocated resources. As demonstrated in the evaluation, our solution overcomes these limitations.

7 Conclusion

In this paper, we have proposed a resource-level SLA metric for specifying fine-grain QoS guarantees regarding the computing capacity of a provider. This metric overcomes the limitations of traditional CPU-related metrics. In particular, it takes into account if services are really using their allocated resources. This allows the provider to implement dynamic resource provisioning, allocating to the different services only the amount of CPU they need. In this way, better resource utilization in the provider can be achieved. In addition, dynamic resource provisioning can be accomplished while avoiding the fake SLA violations that could happen with other metrics when the customers' services do not use all the resources they have allocated. Finally, it can be used in heterogeneous machines, since it is based on Amazon's ECUs. As demonstrated in the evaluation, using this metric reduces the number of SLA violations, since it only detects those that are a real responsibility of the provider.

Although we have derived a metric for managing CPU, the idea of considering the real resource usage of the services can be used with other resources such as the memory or the network. This is part of our future work. In addition, we are working on mechanisms for supporting the inference of the resource requirements of a service to fulfill a given service-level metric.

Acknowledgments

This work is supported by the Ministry of Science and Technology of Spain and the European Union (FEDER funds) under contract TIN2007-60625 and grant AP2008-0264, and by the Generalitat de Catalunya (contract 2009-SGR-980).

References

1. The Grid Workloads Archive, `http://gwa.ewi.tudelft.nl`
2. 3Tera SLA, `http://www.3tera.com/News/Press-Releases/Recent/3Tera-Introduces-the-First-Five-Nines-Cloud-Computing.php`
3. Abrahao, B., Almeida, V., Almeida, J., Zhang, A., Beyer, D., Safai, F.: Self-Adaptive SLA-Driven Capacity Management for Internet Services. In: 10th IEEE/IFIP Network Operations and Management Symposium (NOMS 2006), Vancouver, Canada, April 3-7, pp. 557–568 (2006)
4. Amazon EC2, `http://aws.amazon.com/ec2/`
5. Andrieux, A., Czajkowski, K., Dan, A., Keahey, K., Ludwig, H., Pruyne, J., Rofrano, J., Tuecke, S., Xu, M.: Web Services Agreement Specification (WS-Agreement). In: Global Grid Forum GRAAP-WG, Draft (August 2004)

6. Aneka Software, http://www.manjrasoft.com/products.html
7. Appleby, K., Fakhouri, S., Fong, L., Goldszmidt, G., Krishnakumar, S., Pazel, D., Pershing, J., Rochwerger, B.: Oceano - SLA-based Management of a Computing Utility. In: Symposium on Integrated Network Management (IM 2001), Seattle, WA, USA, May 14-18, pp. 855–868 (2001)
8. Bennani, M., Menasce, D.: Resource Allocation for Autonomic Data Centers using Analytic Performance Models. In: 2nd International Conference on Autonomic Computing (ICAC 2005), Seattle, USA, June 13-16, pp. 229–240 (2005)
9. EU BREIN project, http://www.eu-brein.com
10. Ching, A., Sacks, L., McKee, P.: SLA Management and Resource Modelling for Grid Computing. In: London Communications Symposium (LCS 2003), London, UK, September 8-9 (2003)
11. van Dorst, W.: BogoMips mini-Howto. Tech. rep., Clifton Scientific Text Services V38 (March 2006)
12. Amazon EC2 Service Level Agreement, http://aws.amazon.com/ec2-sla/
13. Fitó, J.O., Goiri, I., Guitart, J.: SLA-driven Elastic Cloud Hosting Provider. In: 18th Euromicro Conference on Parallel, Distributed and Network-based Processing (PDP 2010), Pisa, Italy, February 17-19, pp. 111–118 (2010)
14. GoGrid Service Level Agreement, http://www.gogrid.com/legal/sla.php
15. Goiri, I., Julia, F., Ejarque, J., Palol, M., Badia, R., Guitart, J., Torres, J.: Introducing Virtual Execution Environments for Application Lifecycle Management and SLA-Driven Resource Distribution within Service Providers. In: 8th IEEE International Symposium on Network Computing and Applications (NCA 2009), July 9-11, pp. 211–218 (2009)
16. Keller, A., Ludwig, H.: The WSLA Framework: Specifying and Monitoring Service Level Agreements for Web Services. Journal of Network and Systems Management 11(1), 57–81 (2003)
17. Lodi, G., Panzieri, F., Rossi, D., Turrini, E.: SLA-Driven Clustering of QoS-Aware Application Servers. IEEE Transactions on Software Engineering 33(3), 186–197 (2007)
18. Menasce, D., Casalicchio, E.: Quality of Service Aspects and Metrics in Grid Computing. In: Computer Measurement Group Conference (CMG 2004), Las Vegas, USA, December 5-10, pp. 521–532 (2004)
19. Raj, H., Nathuji, R., Singh, A., England, P.: Resource Management for Isolation Enhanced Cloud Services. In: 2009 ACM Cloud Computing Security Workshop (CCSW 2009), Chicago, IL, USA, November 13, pp. 77–84. ACM, New York (2009)
20. Sahai, A., Graupner, S., Machiraju, V., van Moorsel, A.: Specifying and Monitoring Guarantees in Commercial Grids through SLA. In: 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid 2003), Tokyo, Japan, May 12-15, pp. 292–299 (2003)
21. Skital, L., Janusz, M., Slota, R., Kitowski, J.: Service Level Agreement Metrics for Real-Time Application on the Grid. In: Wyrzykowski, R., Dongarra, J., Karczewski, K., Wasniewski, J. (eds.) PPAM 2007. LNCS, vol. 4967, pp. 798–806. Springer, Heidelberg (2008)
22. Xen Hypervisor, http://www.xen.org
23. Xu, J., Zhao, M., Fortes, J., Carpenter, R., Yousif, M.: On the Use of Fuzzy Modeling in Virtualized Data Center Management. In: 4th International Conference on Autonomic Computing (ICAC 2007), Jacksonville, USA, June 11-15, p. 25 (2007)

A Framework for Building Intelligent SLA Negotiation Strategies under Time Constraints

Gheorghe Cosmin Silaghi, Liviu Dan Şerban, and Cristian Marius Litan

Babeş-Bolyai University
Str. Theodor Mihali 58-60, 400591, Cluj-Napoca, Romania
{Gheorghe.Silaghi,Liviu.Serban,Cristian.Litan}@econ.ubbcluj.ro

Abstract. In the context of self-management of SLAs, automated and intelligent negotiation solutions for reaching agreements are a hot research topic, especially for self-organizing distributed systems. Previous work regarding SLA negotiation in grids focuses on devising bargaining models where service providers and consumers can meet and exchange SLA offers and counteroffers. Recent developments in agent research further introduce intelligent strategies for contract negotiation. In this paper we build on the latest developments in agent research to provide a generic framework for strategical negotiating of service level values under time constraints. We exemplify the usage of our generic framework by extending the Bayesian learning agent. We prove that a time-constrained negotiation strategy can surpass the baseline learning scheme in a longer-lasting negotiation, because it gives the intelligent learning scheme time to converge and to estimate better the opponent's profile.

1 Introduction

Various distributed environments like grids or peer-to-peer systems organized around service-oriented principles require strong Service Level Agreement management as a key building block. The grid envisions the virtual organization concept to enable collaboration around shared resources and the SLAs control the usage and provision of resource by the third parties [1]. In an open environment with competing stakeholders (like service-oriented peer-to-peer systems), SLAs gain further importance as they control also the accounting part. As envisioned by the SLA@SOI project, SLAs are the key element towards a business ready service-oriented infrastructure empowering the service economy in a flexible and dependable way [2].

Within self-management of SLAs, automation of SLA negotiation is a key point. All major grid/cloud infrastructures provide SLA support within the middleware, implementing solutions for: SLA formation, service provision and monitoring, accounting and even SLA translation. In this context, automated and intelligent negotiation solutions are still a hot research topic, especially for self-organizing distributed systems [3].

J. Altmann and O.F. Rana (Eds.): GECON 2010, LNCS 6296, pp. 48–61, 2010.

In this paper we tackle the subject of automating SLA negotiation within SLA-empowered virtual organization, with competing negotiating parties. Previous work regarding SLA negotiation in grids [4,5] focuses on devising bargaining models where service providers and consumers can meet and exchange SLA offers and counteroffers. Here we introduce intelligent strategies for contracts negotiation, recently developed in agent research [6,7,8,9,10,11,12] and present a framework for strategical negotiating of service level values under time constraints. The main contribution of our framework is its generality, in the sense that it can be adapted to any standard intelligent negotiation strategy that learns some opponent's profile. Time-constrained negotiation is important in a SLA-aware middleware, because, every service broker works in finite time and can not endlessly wait to produce a service allocation.

In section 2 we formally describe the SLA negotiation problem. We start by formalizing the SLA negotiation game in an open virtual organization. Next, we present the state-of-the-art regarding intelligent negotiation strategies and shortly describe the Bayesian-learning agent, further used as a baseline example to develop a time-constrained strategy. Section 3 presents the general framework for building a intelligent time-constrained negotiation strategy. Section 4 presents the experimental results of a particular time-constrained negotiation strategy against several other strategies considered by the literature and section 5 concludes the paper.

2 Background

In this section we will formalize the SLA negotiation setup and will introduce the existing work on negotiation strategies, which constitutes the foundation of our contribution.

2.1 SLAs Formalization

In this subsection we formalize the SLA negotiation setup. We take the formalization presented in [13], as it directly applies to SLA-based virtual organizations and quality-of-services negotiation.

We tackle a virtual environment with service providers and users (consumers). Let $X = \{x_1, x_2, \ldots, x_n\}$ denote the set of all services, with x ranging on X. Let SP denote the set of service providers, with b ranging on SP, and function $S : SP \rightarrow \mathscr{P}(X)$ denoting the services provided by a service provider, where $\mathscr{P}$ represents the power set operator. Let SC denote the set of users (service consumers) of the system, with c ranging on SC.

Each service has associated issues of interest, denoted by set I, which users are interested in negotiating; variable i ranges on I. Function IS represents the set of issues of interest for a service: $IS : X \rightarrow \mathscr{P}(I)$. Given an SLA-based environment, for a given service, users are interested in negotiating issues like the price, penalty and the properties of the service. Our framework covers both functional and non-functional attributes of a service, given that for each

attribute, either it can be defined a finite list of possible options or the service level values are expressed as real numbers.

Negotiation finishes with a contract (the SLA) that is composed of the actual price paid for the service, the penalty and the actual service level values. Considering the formalization proposed in [13], function $O^c : X \times SP \times I \rightarrow \mathscr{R}$ denotes the expectation of user c on the services he uses, where $\mathscr{R}$ denotes the real numbers. Notation $v_{x,i}^{b,c}$ represent the expectation of user c on issue i of service x supplied by provider b, which in fact, are the service level values.

In this paper, we focus on the bilateral SLA negotiation, between a service provider b and an user c for a given service s. We assume that agents are in place to automate this negotiation, and for each negotiation session we *restrict* the available time to T_{max}.

We denote by $U(v) = U_x^{c,b}(v)$ the utility that user c gets by obtaining the actual value $v = (v_{x,i_1}^{b,c}, v_{x,i_2}^{b,c}, \ldots, v_{x,i_n}^{b,c})$ of the consumed service. From now on, as we deal only with a given service, supplied by a given provider and consumed by a specified user, we will drop the letters c, b and x.

We assume that the utility function $U(v)$ can be written by a linear combination of the individual utility functions $U_k(v_k)$ (á la Von Neumann):

$$U(v) = \sum_{k=1}^{n} w_k U_k(v_k), \ with \ \sum_{k=1}^{n} w_k = 1. \tag{1}$$

where $U_k(v_k)$ represents the utility that the consumer obtains by receiving the value $v_k = v_{x,i_k}^{b,c}$ for the issue i_k and $0 \leq w_k \leq 1$ represent the weights measuring the importance of a given issue k for the user.

2.2 Negotiation Strategies - Related Work

When a provider (seller) negotiates with a consumer (buyer), both parties are interested in obtaining those contract values $v = (v_{i_1}, v_{i_2}, \ldots, vi_n)$ that maximize their utility functions $U(v)$. We should note that negotiation implies a trade-off between the negotiating parties, as their utility functions are not the same.

In Grid computing, strategic negotiation models for automated negotiation were proposed [4]. Besides this simple bilateral approach, concurrent negotiation setups were also investigated [7], considering possible out-side options available for both parties [5]. In the simple bilateral negotiation setup, learning strategies were investigated and tested. Li & Yahyapour [4] considers Q-Learning [14] applied directly for SLA negotiation with time-decreasing utility functions. Models for SLA negotiation in Grids were neither concerned on the optimality of the solution or the strengths of the negotiation strategy. These models emphasize of the automation and the gains that can be obtained in comparison with a non-automated and zero intelligence solution.

Agent research focuses more strongly on the formal aspects of the negotiation and proposes more elaborated negotiation strategies. Negotiating agents were developed for the bilateral setup, including the following ones. The zero-intelligent agent [15] investigates the baseline behavior of a random agent. Jonker & Treur

[16] presents an agent devised in-line with the Belief-Desire-Intention architecture of the mid-90's agents. Based on this architecture, Jonker et al. [10] further investigates whether information disclosure during negotiation is valuable for the agents and shows that even with a limited amount of preference information revealed, agents can still obtain significant joint gains in the outcome. Within the same BDI architecture, Lopes et al. [8] presents a generic framework to test tactics during a negotiation, such as: starting high and conceding slowly, starting reasonable and conceding moderately, respectively starting low and conceding rapidly. We consider that our framework subscribes to the first mentioned negotiation tactic of [8] (starting high and conceding slowly).

Raeesy et. al [9] applies the fuzzy logic to model the negotiation agents and to describe their strategic behavior. A Bayesian learning framework is introduced by Hindriks & Tykhonov [11] to approximate the user profiles. Negotiation with the help of Q-Learning is investigated in [12,17]. Rather than learning the opponent's profile, Faratin et. al [6] investigates the optimal strategy to perform trade-offs when proposing new bids.

We should insist on the difficulty to practically compare these negotiation solutions. Each author of a negotiation strategy devised it within some specific setups and no strategy was ever compared against the others with the same environmental assumptions. To overcome these problems, the Genius negotiator[1] [18] was developed at the Delft University of Technology, supplying a unified environment for negotiation strategies comparison. However, up to date, only some of the strategies presented above [10,11,15] are implemented in Genius. To ease the comparison of our negotiation strategy with the existing literature, we implemented it in Genius (see section 4).

In all these papers from agent research, the focus is on learning the opponent's profile and preferences - time restriction is not a major concern, and, given this, to devise negotiation strategies for mutual benefit of both parties. In our paper we insist on devising a time-based strategy, that in conjunction with a learning procedure, brings further advantage and gains from the negotiation process. Furthermore, the negotiation under time constraints makes the strategy suitable for the practical requirements of a SLA-based self-organizing system.

2.3 The Bayesian-Learning Agent

In general, at each negotiation step, a learning-based agent performs two actions:

- analyze the incoming opponent's proposal and update the opponent's profile
- select and propose the next bid

Learning the opponent's profile consists mainly of approximating its $U(v)$ utility function. While we assume that the utility function $U(v)$ is a linear combination (see eq. 1), the opponent profile is made of the individual utility functions $U_k(v_k)$ and the weights w_k, for all $k = \overline{1,n}$.

[1] http://mmi.tudelft.nl/negotiation/index.php/Genius. We thank Prof. Catholijn Jonker from Delft University of Technology Netherlands that provided us a comprehensive introduction to Genius and Dmytro Tykhonov for technical support.

As the search space for the opponent profile is infinite, different learning models propose various techniques to reduce it and to make it computer tractable via simplification. Because, in our paper we employed the Bayesian learning [11], we will shortly present below the main characteristics of this technique.

Bayesian learning does not precisely learn the weights w_k. While the literature [6] argues that learning the ranks of the weights is typically sufficient to significantly increase the efficiency of the outcome, the search space for the weights reduces to the $n!$ possible rankings of the preferences for the n issues.

For each individual utility function $U_k(v_k)$, the agent preferences for the issue values can be encoded starting from three basic function shapes:

- downhill shape: minimal issue values are preferred over other issue values and the evaluation of issue values decreases linearly when the value of the issue increases;
- uphill shape: maximal issue values are preferred over other issue values and the evaluation of issue values increases linearly when the value of the issue increases;
- triangular shape: a specific issue value somewhere in the issue range is valued most and and evaluations associated with issues to the left and right of this issue value linearly decrease

A function with a unique extreme point on an interval can be approximated with a linear combination of *basis functions*[2] from the three types presented above. Therefore, given that the domain of an issue k has m_k values, one can draw m_k functions with the maximal issue value in each of the m_k values and linearly combine these functions to obtain the individual utility function U_k. This approach is very suitable for discrete issue domains, while continuous domains might be discretized (i.e. split in m equal subintervals). Thus, for an issue k, the search for the individual utility function U_k is in fact reduced to learning some numeric values $0 \leq p_i \leq 1$, $i = \overline{1, m_k}$, which allows to compose U_k as a linear combination of triangular functions.

From the discussion above, we note that learning the opponent profile means selecting the proper ranking of weights and assigning some probabilistic values for each triangular valuation function for each issue value, for all issues. Therefore, in the Bayesian learning setup, a user profile is a matrix that associates probabilities for each hypothesis $h_j \in H$, where $H = H_w \times H_1 \times H_2 \times \ldots H_n$ is the hypothesis space. One row of this matrix represents the probabilities $(p_w, p_1, \ldots, p_n)$ associated for a hypothesis h, and the matrix has $M = n! \times m_1 \times m_2 \times \cdots \times m_n$ lines (the total number of possible hypotheses), .

[2] In mathematics, a basis function is an element of a particular basis for a function space. Every function in the function space can be represented as a linear combination of basis functions, just as every vector in a vector space can be represented as a linear combination of basis vectors (G. Strang, "Wavelets", American Scientist, Vol. 82, 1994, pp. 250-255.).

Bayesian learning starts with some random initialization of the agent profile matrix. At each step, this matrix is updated using the Bayes rule. If b_t is the newest received bid, then

$$P(h_j|b_t) = \frac{P(h_j)P(b_t|h_j)}{\sum_{k=1}^{M} P(h_k)P(b_t|h_k)} \tag{2}$$

Hindriks & Tykhonov [11] shows how to compute eq. 2, given the setup described above. Is worth to note that if the opponent agent is a rational one, after a big number of proposed bids, the agent profile matrix converges.

The second action that a negotiating agent performs in each step is *to select and propose the next bid*. For each possible own bid $b = (v_1, v_2, \ldots, v_n)$ the agent can compute the estimated utility of this bid for the opponent:

$$U(b) = \sum_{j=1}^{M} P(h_j) \sum_{k=1}^{n} w_k U_k(v_k) \tag{3}$$

A smart strategy is to select and propose that bid that maximizes the utility U_b of the opponent, while the own utility U_{own} to stay as close as possible to the own target negotiation objective τ:

$$b = \underset{b' \in \{x || U_{own}(x) - \tau| \leq \delta\}}{\arg\max} U(b') \tag{4}$$

Programming a Bayesian learning agent is not easy, as the programmer must tackle the complexity of the underlying mathematics and also, the agent should react in a *reasonable time*. Both actions to be performed at each step include high running complexity, and code and computation optimizations are required to make the agent responsive. Hindriks & Tykhonov [11] proposes some optimizations and shows the power of the Bayesian agent against other negotiating strategies, including trade-off [6] and ABMP [10].

We note several drawbacks of the Bayesian agent:

- it does not learn the negotiation strategy of the opponent, as, at every step, it updates the opponent profile only with the information about its *last* bid, never considering the succession of the opponent's bids;
- the agent is highly dependent on the initialization step. At the beginning of the negotiation, the opponent profile is very weak and the agent will propose highly suboptimal bids. Thus, if the negotiation finishes in the first rounds, the Bayesian agent is highly likely to lose.
- as devised by [11], the Bayesian agent is a concession agent, it linearly decreases the predicted utility value for the opponent over the time. Thus, the agent has more chances to finalize the negotiation in few rounds, which raises the drawback presented just above.

To overcome these drawbacks and to improve the agent as much as possible, suited for a negotiation under time constraints, we propose the framework presented in the following section. We should note that this framework is general enough to be valid in relation with other profile learning strategies.

3 The Time-Constrained Negotiation Strategy

In what follows we present our strategy for negotiation under time constraints. We assume an automated bilateral negotiation, with exchange of bids between the user agent (the consumer) and the service provider agent. A bid b is in fact a SLA proposal, performed at time t. Negotiation should finish before time T_{max}. If the user and the provider are not able to establish an agreement before T_{max}, they gain nothing from the negotiation game. We insist on a single negotiation game and we do not consider several negotiation sessions in a row.

The agent designed in this section is a sort of learning-agent, presented in subsection 2.3, performing two actions at each step: (i) updates the opponent's profile and (ii) select and propose the next bid. Our agent can play both roles, either user (buyer / consumer), or service provider (seller). Although we design our agent for the Bayesian learning setup, we do not restrict the agent design to this specific learning scheme, nor do we restrict the agent for a particular negotiation domain.

The core of our negotiation strategy consists in adapting the agent behavior to the duration of the negotiation. Usually, a learning scheme takes time to converge to some consistent opponent's profile, therefore the agent should consider to allow several bilateral bids exchange rounds without much own utility concession and concede only when the total negotiation time T_{max} is about to elapse.

Our strategy is to divide the negotiation time $[0,T_{max}]$ in k subintervals $I_1, I_2, \ldots, I_k$, $I_1 = [0,t_1]$, $\ldots$, $I_k = (t_{k-1},T_{max}]$ and let the agent play different strategies on each subinterval. During all negotiation time, the agent applies the baseline learning scheme in order to estimate the opponent's profile. The strategies played on each subinterval $I_1, I_2, \ldots, I_k$ differ on the step when the agent selects and proposes the next bid such as follows:

- Conditions C_1 for selecting the next bid in subinterval I_1 are the strongest, the agent performing very few concessions at the beginning of the negotiation;
- Conditions C_{t-1} for selecting the next bid in the subinterval I_{t-1} are stronger than the conditions C_t for selecting the next bid in the subinterval I_t, for every $t = \overline{2,k}$;
- Conditions C_k for selecting the next bid in the last subinterval I_k are the baseline conditions of the learning scheme

It results that $C_1 \supset C_2 \supset \cdots \supset C_k$.

In our case, we selected $k = 3$ and divided the interval $[0,T_{max}]$ on three subintervals:

- interval I_1: from [0,t_1], the agent performs only very small concessions.
- interval I_2: from (t_1,t_2], the agent performs like in I_1, but we relax some conditions for selecting the next bid
- interval I_3: from $(t_2, T_{max}]$, the agent concedes like in the standard setup imposed by the baseline learning agent. Our agent will concede down to a reservation utility, although the agent should be better off with any gains from the negotiation if the total negotiation time elapses without any agreement.

When designing our agent we properly selected the interval limits t_1 and t_2 by experimentation and devised the conditions for when and how to perform concessions.

We introduced the second interval I_2, because we noticed that, without this interval, the agent switches very abrupt from a powerful agent that does not concede to an agent that concedes very quickly. If a smart agent is considered as the opponent, this agent can simply learn this behavior and our agent will never win in this case. By performing the transition between the strong-holding agent behavior of I_1 to the standard behavior of I_3 via some "mixed" behavior in I_2, we harden the job of an opponent to learn our strategy.

The reader can also note that on I_3, at the end of the negotiation, we do not accept an agreement below our reservation utility u_r. This can be: a) the utility of the bid that maximizes the estimated opponent's utility on the whole domain; or b) the utility given by the first offer of the opponent; or c) another of its past offers. Any of these choices should give us an acceptable positive utility, ensuring as well that the opponent will accept it, thus avoiding the negotiation to end without an agreement.

We note that the time-constrained negotiation framework obtained by the division of the negotiation interval in k subsets is general enough to be particularized to a specific learning procedure, considering the more or less available conditions for next bid selection. In general, we recommend setting the subinterval limits $t_1, \ldots, t_{k-1}$ as far as possible towards T_{max}, making the agent as inflexible as possible till very close to the ending time.

The particular application presented in here is an opponent model based agent, which means that it estimates the opponent's utility and use the estimation to propose a bid with reasonable utility for the latter. The learning method used to estimate the opponent's preferences is Bayesian. The preferences' estimation allows the evaluation of the opponent's utility for any possible bid. Remember from Section 2.3 the general form of a decision of our agent, at a given point in time; that is, to propose a bid that maximizes the utility U_b of the opponent, while its own utility U_{own} varies in an interval $(u_{min}, u_{max}]$:

$$b = \underset{b' \in \{x | U_{own}(x) \in (u_{min}, u_{max}]\}}{\arg\max} U(b') \tag{5}$$

The method by which $(u_{min}, u_{max}]$ is updated on a given time interval, generates less, or more concessions from our agent, for the time interval in discussion. The baseline for the update of $(u_{min}, u_{max}]$ is the following: $u_{min} = u_{max} - \delta$, and

at each offer, if our bid is refused, then the upper bound of the interval (and implicitly the lower bound) moves to the left. The starting value u_{max} of the upper bound is represented by the highest utility of the agent, and then it is sequentially moved to the left. Notice that u_{max} takes the place of the targeted utility τ from eq. 4, from which we accept a deviation of a maximum δ.

Next we explain in detail the strategies of our agent (i.e. the decision rules to offer a bid plus the method of updating $(u_{min}, u_{max}]$) for each of the three particular time intervals considered by our application:

Interval I_1 (btw. 0-85% of the total time): Supposing that b_t is the last bid proposed by the opponent, then at step $t+1$ the own agent will propose a bid b_{t+1} that not only maximizes the opponent's utility, but also ensures that our agent does not attain an utility lower than the one implied by the opponent's offered bid b_t. Mathematically:

$$u_{min} = \max\{u_{max} - \delta, U_{own}(b_t)\} \tag{6}$$

Notice that on this time interval, the negotiation game may end only with the opponent accepting an offered bid; our agent would never accept a bid offered by the opponent.

Interval I_2 (btw. 85-95% of the total time): The conditions for selecting our next bid are relaxed to the usual $u_{min} = u_{max} - \delta$. Our agent accepts the offer b_t only if $U_{own}(b_t) > U_{own}(b_{t+1})$.

Interval I_3 (btw. 95-100% of the total time): The concessions our agent is ready to make increase dramatically. Establish a reservation utility u_r for our agent.

The update of the lower limit will be given mathematically by:

$$u_{min} = \min\{\max(u_r, u_{max} - \delta), e^{c+d(T_{max}-t)}\} \tag{7}$$

where T_{max} is the time constraint of the negotiation, $c = \ln(u_r)$, $d = \frac{\ln(\frac{u_{0min}}{u_r})}{(1-0.95)\cdot T_{max}}$, and u_{0min} is the last lower bound before entering in the time interval I_3.

Notice that for $t = T_{max}$, then

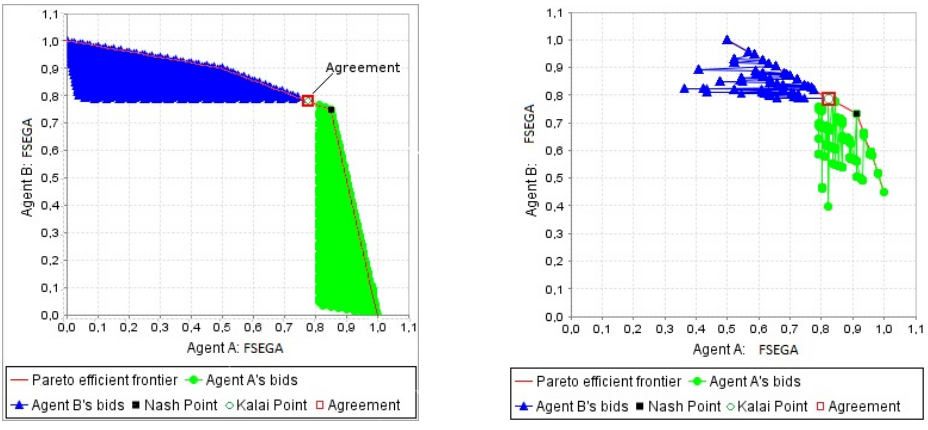
$$e^{c+d(T_{max}-t)} = u_r \tag{8}$$

which is the minimum of the given exponential function. For $t = 0.95 \cdot T_{max}$, then

$$e^{c+d(T_{max}-t)} = u_{0min} \tag{9}$$

which is the maximum of the given exponential function. The rest of the conditions for selecting our next bid, or accepting an offered bid, remain the same as before.

It is straightforward to prove that the set of conditions for selecting the next bid are less and less restrictive as we move from interval I_1 to the interval I_3 (provided that δ is not too big). Therefore, the requirements of the general setup are fulfilled.



(a) The SON domain (b) The Zimbabwe England domain

Fig. 1. AgentFSEGA against itself

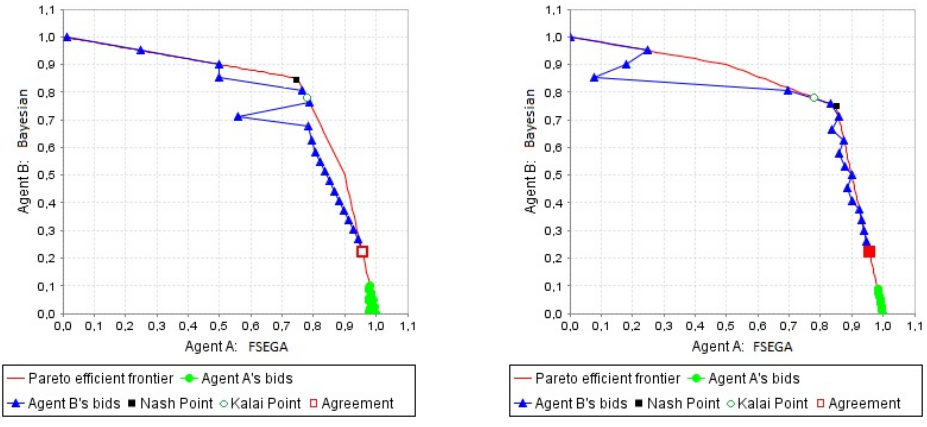
4 Experimental Results

We implemented a time-constrained negotiation strategy with 3 subintervals, using the Bayesian learning scheme as baseline in the Genius negotiator [18].

We tested our negotiation strategy against the ABMP [10] and the standard Bayesian [11] agents, both publicly available in the Genius negotiator. We named our negotiation strategy AgentFSEGA.

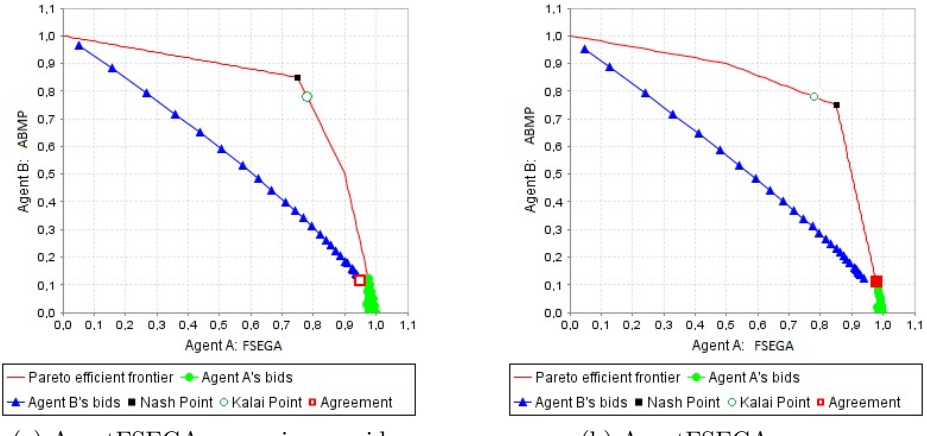
We selected as testing domain the service oriented negotiation [6] (SON), which fully complies with the SLA formalization presented in section 2.1. The service described in the SON domains has four issues of interest, each having 30 possible values. Thus, there are 810000 total number of possible agreements. SON domain is huge, in the sense that are a lot of possible agreements and requires a very scalable agent. We also tested our agent on another domain, namely the Zimbabwe England domain [19], which has only 576 possible agreements, but with a stronger opposition between payoffs received by the negotiating parties (there are fewer bids mutually acceptable for both parties, thus harder to get an agreement). We set the time limit for a negotiation game to 3 minutes.

On figure 1 we depict AgentFSEGA negotiating against itself on both domains. On each axes, we depict the utilities for the negotiating agents. On both figures we depicted the Pareto frontier (with simple line), service provider's bids (with strong circled line) and the opponent's bids (line with triangles). The empty square point represents the agreement, the black small square represents the Nash equilibrium point, while the empty circle represents the Kalai-Smorodinsky point. (The Nash point represents the agreement that maximizes the product of utilities. The Kalai-Smorodinsky solution represents the point where the hypothetic line from 0 towards the point where both players get the maximum payoff intersects the Pareto frontier.)



(a) AgentFSEGA as service provider (b) AgentFSEGA as user

Fig. 2. AgentFSEGA against Bayesian on the SON domain



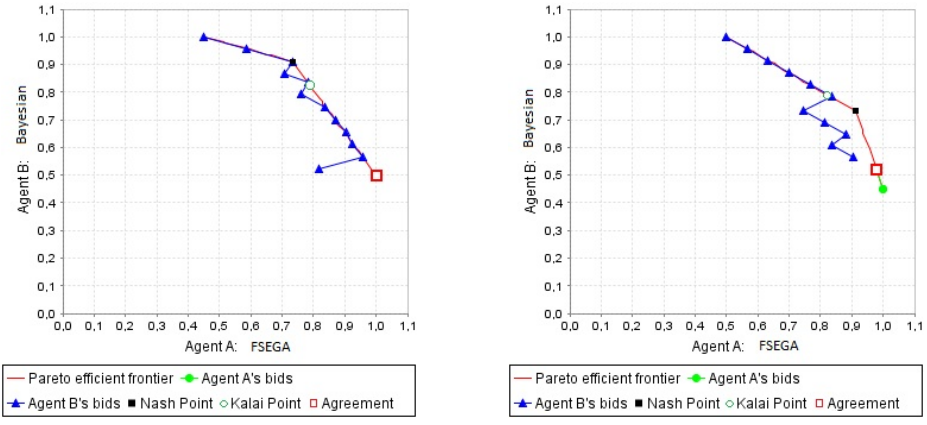
(a) AgentFSEGA as service provider (b) AgentFSEGA as user

Fig. 3. AgentFSEGA against ABMP on the SON domain

We note that the agreement is established on the Kalai Smorodinsky point, which maximize both agents payoffs, thus full cooperation is induced by the game. To reach the agreement, about 6000 couples of offers/counteroffers are exchanged. This shows that AgentFSEGA reasons fast-enough to be suitable for practical deployments.

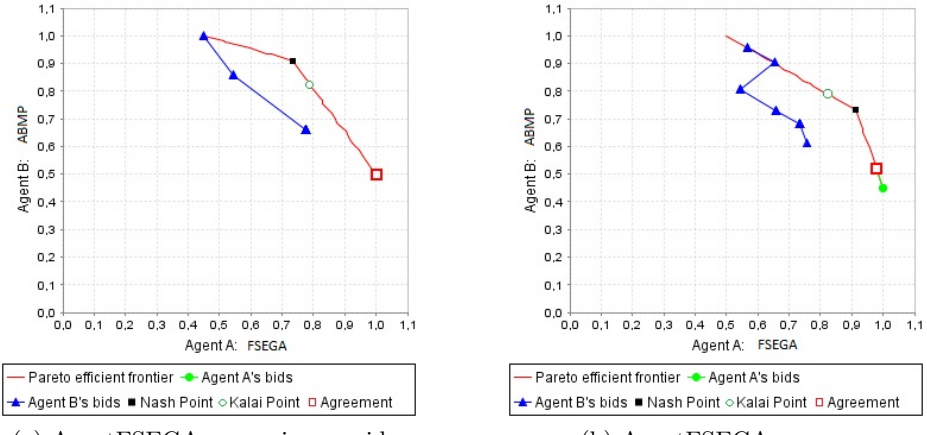
Concluding, if both the user and the provider negotiate with similar intelligence, both revenues are maximized - therefore is valuable to equip a SLA-based environment with such sort of intelligence. This finding also validates the correctness of our agent implementation.

Figures 2 and 3 presents the result of AgentFSEGA playing as a user or provider against the Bayesian and ABMP strategies on the SON domain. This



(a) AgentFSEGA as service provider (b) AgentFSEGA as user

Fig. 4. AgentFSEGA against Bayesian on the England - Zimbabwe domain



(a) AgentFSEGA as service provider (b) AgentFSEGA as user

Fig. 5. AgentFSEGA against ABMP on the England - Zimbabwe domain

experiment models the case of an SLA-based environment with competing stakeholders of different intelligence. We note that in both cases, AgentFSEGA does not concede at the beginning of the negotiation.

Similar findings come out from the England - Zimbabwe domain (figures 4 and 5), even if the agreement point is harder to show out, i.e. higher number of rounds are needed to reach the agreement.

Both experiments show that the time-constrained agent surpass the baseline agent and other non-time constrained strategies. This emphasize our judgment that letting the learning strategy to converge is worth to consider, given that some finite T_{max} negotiation time is available.

5 Conclusions and Future Work

This paper tackles the automatic SLA negotiation in virtual environments with competing resource owners and consumers. While previous work on SLA negotiation in grids focuses on providing negotiation models in the bargaining game, in this paper we introduce a framework for building automatic negotiation strategies under time constraints. We build our work on the latest results concerning negotiation strategies developed in agent research and specifically, on the Bayesian learning agent. To instantiate our general framework for intelligent negotiation strategies under time constraints, we further extend and modify the Bayesian agent.

We show that the time-constrained negotiation framework can help in building much robust and better negotiation strategies for service level values establishment, given that time is a key resource in every SLA negotiation in grid/cloud middleware. By presenting the implementation of the Bayesian agent in the proposed framework, we present a valuable experience about how to extend am opponent modeling-based strategy to cope the time restrictions. As a further work, other intelligent strategies need to be adapted in the presented framework and further testing is required, even on more complex and difficult negotiation domains.

Acknowledgment

This work is supported by the Romanian Authority for Scientific Research under project PNII IDEI_2452.

References

1. Foster, I., Kesselman, C., Tuecke, S.: The anatomy of the grid: Enabling scalable virtual organizations. Int. J. High Perform. Comput. Appl. 15(3), 200–222 (2001)
2. Sla@soi - overview. SLA@SOI consortium (2008), document available at, http://sla-at-soi.eu/publications/
3. Sun, Y.L., Edmonds, A., Kennedy, J., Bayon, V., Hadalin, P.: Sla-aware resource management. In: Proceedings of the Service Level Agreements in Grids Workshop at Grid 2009. CoreGrid Springer series (2010) (to appear)
4. Li, J., Yahyapour, R.: A strategic negotiation model for grid scheduling. International Transactions on Systems Science and Applications 1(4), 411–420 (2006)
5. Li, J., Sim, K.M., Yahyapour, R.: Negotiation strategies considering opportunity functions for grid scheduling. In: Kermarrec, A.-M., Bougé, L., Priol, T. (eds.) Euro-Par 2007. LNCS, vol. 4641, Springer, Heidelberg (2007)
6. Faratin, P., Sierra, C., Jennings, N.: Using similarity criteria to make issue tradeoffs in automated negotiations. Artificial Intelligence 142(2) (2002)
7. Nguyen, T.D., Jennings, N.R.: A heuristic model for concurrent bi-lateral negotiations in incomplete information settings. In: IJCAI 2003: Proceedings of the 18th International Joint Conference on Artificial Intelligence, pp. 1467–1469. Morgan Kaufmann, San Francisco (2003)

8. Lopes, F., Mamede, N., Novais, A., Coelho, H.: Negotiation strategies for autonomous computational agents. In: ECAI 2004: Proceedings of the 16th European Conference on Artificial Intelligence, pp. 38–42. IOS Press, Amsterdam (2004)
9. Raeesy, Z., Brzostowski, J., Kowalczyk, R.: Towards a fuzzy-based model for human-like multi-agent negotiation. In: Proceedings of the 2007 IEEE/WIC/ACM International Conference on Intelligent Agent Technology, pp. 515–519. IEEE Computer Society, Los Alamitos (2007)
10. Jonker, C.M., Robu, V., Treur, J.: An agent architecture for multi-attribute negotiation using incomplete preference information. Autonomous Agents and Multi-Agent Systems 15(2), 221–252 (2007)
11. Hindriks, K., Tykhonov, D.: Opponent modelling in automated multi-issue negotiation using bayesian learning. In: AAMAS 2008: Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems, IFAAMAS, pp. 331–338 (2008)
12. Jian, L.: An agent bilateral multi-issue alternate bidding negotiation protocol based on reinforcement learning and its application in e-commerce. In: ISECS 2008: Proceedings of the 2008 International Symposium on Electronic Commerce and Security, pp. 217–220. IEEE Computer Society, Los Alamitos (2008)
13. Silaghi, G.C., Arenas, A.E., Silva, L.M.: A utility-based reputation model for service-oriented computing. In: Priol, T., Vanneschi, M. (eds.) Towards Next Generation Grids, CoreGRID Symposium, Rennes, France, August 27-28, pp. 63–72. Springer, Heidelberg (2007)
14. Watkins, C.J.C.H., Dayan, P.: Q-learning. Machine Learning 8(3-4), 279–292 (1992)
15. Gode, D.K., Shyam, S.: Allocative efficiency of markets with zero-intelligence traders: Market as a partial substitute for individual rationality. The Journal of Political Economy 101(1), 119–137 (1993)
16. Jonker, C.M., Treur, J.: An agent architecture for multi-attribute negotiation. In: IJCAI 2001: Proceedings of the 17th International Joint Conference on Artificial Intelligence, pp. 1195–1201. Morgan Kaufmann, San Francisco (2001)
17. Dearden, R., Friedman, N., Russel, S.: Bayesian q-learning. In: Proceedings of the Fifteenth National Conference on Artificial Intelligence, pp. 761–768 (1998)
18. Hindriks, K., Jonker, C.M., Kraus, S., Lin, R., Tykhonov, D.: Genius: negotiation environment for heterogeneous agents. In: AAMAS 2009: Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems, IFAAMAS, pp. 1397–1398 (2009)
19. Lin, R., Oshrat, Y., Kraus, S.: Investigating the benefits of automated negotiations in enhancing people's negotiation skills. In: AAMAS 2009: Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems, IFAAMAS, pp. 345–352 (2009)

Agent-Based Simulations of the Software Market under Different Pricing Schemes for Software-as-a-Service and Perpetual Software

Juthasit Rohitratana and Jörn Altmann

Technology Management, Economics, and Policy Program
Department of Industrial Engineering, College of Engineering, Seoul National University
599 Gwanak-Ro, Gwanak-Gu, Seoul 151-744, South-Korea
juthasit@temep.snu.ac.kr, jorn.altmann@acm.org

Abstract. In this paper, we present agent-based simulations that model the interactions between software buyers and vendors in a software market that offers Software-as-a-Service (SaaS) and perpetual software (PS) licensing under different pricing schemes. In particular, scenarios are simulated, in which vendor agents dynamically set prices. Customer (or buyer) agents respond to these prices by selecting the software license scheme according to four fundamental criteria using Analytic Hierarchy Process (AHP) as decision support mechanism. These criteria relate to finance, software capability, organization, and vendor. Three pricing schemes are implemented for our simulations: derivative-follower (DF), demand-driven (DD), and competitor-oriented (CO). The results show that DD scheme is the most effective method but hard to implement since it requires perfect knowledge about market conditions. This result is supported through a price sensitivity analysis.

Keywords: Software-as-a-Service pricing, perpetual software pricing, agent-based simulation, Analytic Hierarchy Process (AHP), dynamic pricing, decision support.

1 Introduction

In the real world, pricing schemes are various, based on information that vendors can obtain. Software vendors, which try to maximize profit, might use information about customers, competitors, and market demand. Detailed information on customer software selection criteria helps software vendors to design accurate pricing schemes. In addition, software vendors try to obtain a large market share, using network effects and customer lock-in mechanisms, especially if the product differentiation is narrow [10]. However, software vendors, who obtain little information about customers and competitors, can still set their prices based on product development and delivery costs [11]. Simple pricing such as cost-plus and derivative-follower are also appropriated pricing schemes for these software vendors [16].

This paper examines the software market in the situation if SaaS vendors and perpetual software (PS) vendors implement different pricing schemes. For this purpose, we develop agent-based simulations on dynamic software pricing, focusing on SaaS

J. Altmann and O.F. Rana (Eds.): GECON 2010, LNCS 6296, pp. 62–77, 2010.

and perpetual software licensing models. Each vendor agent attempts to maximize its profit by applying one of three pricing schemes: demand-driven (DD), competition-oriented (CO), or derivative-follower (DF) pricing scheme [12]. Customer agents compare (SaaS and perpetual) software application offers made by vendors, using the Analytic Hierarchy Process (AHP) as decision support mechanism. The criteria used for AHP are based on previous software adoption literature. The data used for describing organizations and their IT infrastructures is also based on the result of a literature survey. Software products/services information is collected from offerings of SaaS and perpetual software products in the real-world market.

The simulation results demonstrate how the price impacts the decision making and how different pricing schemes influence the success of the software vendors in the market. It illustrates the ratio of SaaS and perpetual software vendors in the software market. In particular, the results show that SaaS vendors should implement the DF scheme against competitors with a dynamic pricing scheme, but this scheme is not recommended against competitors offering flat rates. For perpetual software vendors, the DF scheme is also effective against the other pricing schemes. The DD scheme generally reaches the lowest price and captures market share. The CO scheme is also good for capturing market share and against competitor with flat rates, but is not doing well if competitors' prices are low. The overall results show that DD scheme is the most effective method but hard to implement. It requires perfect knowledge about the market conditions. These results are supported through a sensitivity analysis.

The remainder of this paper is organized as follows. In section 2, previous research on software selection and dynamic pricing are discussed. Section 3 describes the proposed simulation model, its agents (customer agent and vendor agent), the decision making model, the software selection criteria, and the pricing schemes. Section 4 presents simulation results about the three different pricing schemes of vendor agents. Finally, section 5 concludes the paper.

2 Software Selection and Dynamic Pricing

2.1 Software Selection

Many decision support tools have been suggested for making resource purchases on the Internet [17-18, 20]. In our simulation, customer agents make decisions on choosing a SaaS or perpetual software licensing model. The Analytic Hierarchy Process (AHP) technique, which has been developed by Saaty [13], is chosen as the decision making mechanism.

The AHP method has been implemented in many software selection decision making algorithms. Godse and Mulik applied the AHP technique for the software evaluation process [6]. They also identified the criteria for SaaS adoption: functionality, architecture, usability, vendor reputation, and cost. Lai et al. adopted the AHP method for selecting multimedia authorizing system [7]. Colombo and Francalanci implemented AHP for selecting CRM packages, using criteria based on architectural, functional, and cost requirements [4]. Rohitratana and Altmann aggregated both customer and vendor criteria and used these criteria in their AHP-based decision making model [10]. Mulebeke and Zheng implemented AHP for selecting pen production software, using Web-HIPRE, an AHP software that is available online [8].

2.2 Dynamic Pricing

Dynamic pricing techniques have been implemented to determine price equilibrium, stability, and demand changes [10, 12, 14-16, 19]. Matsuda and Whang studied pricing mechanisms for system administrators to maximize the net value of a network system [14]. They characterized the equilibrium and stability conditions under three dynamic pricing schemes. Altmann et al. investigated the demand change under different pricing schemes [19].

In a simulation of Kephart et al., intelligent agents called *pricebots* automatically update prices of products, using a dynamic pricing algorithm that responds to market condition [12]. Responding to the prices changed by vendor agents, another agent type called *shopbot* compares product prices, ranks products based on customer criteria, and then makes purchase decisions. Dasgupta and Hashimoto applied collaborative filtering to pricebots. After examining customer criteria, they calculate the maximum-profit price through a dynamic pricing algorithm [15].

In another research paper [10], a pricing algorithm uses knowledge about customer preferences. Because of that, the algorithm, which is referred to as a *demand-driven* algorithm, achieves profit maximization. If vendor agents gain perfect knowledge of competitors' prices, the *competition-oriented* pricing algorithm can be implemented [10]. This algorithm only changes the price, if the competitor changes the price. To capture the fact that perfect market knowledge is almost impossible to obtain in the real world, vendors with limited knowledge about customers and competitors can implement the *derivative-follower* algorithm [12]. Using this algorithm, vendor agents incrementally increase (decrease) prices until the profit drops. Then, the prices are decreased (increased), i.e., follow the opposite pricing direction. Dasgupta and Das developed a simulation for pricebots that focuses on the derivative-follower algorithm [16]. They proposed an algorithm that enhances derivative pricing by re-estimating the price-profit relationship for vendor agents in each time period.

3 The Proposed Simulation Model

3.1 Simulation

In our simulation, there are two types of agents: customer agents and software vendor agents. The software vendor agents are categorized into two sub-types: perpetual software vendor agents, and SaaS vendor agents.

After the simulation has started, each vendor agent establishes a price using a dynamic pricing algorithm. Customer agents respond to the prices by calculating scores for each software product option. All software options are ranked based on the scores obtained through a decision support model. Then, the customer agents select the software option with the highest score. All choices are recorded during the simulation run for later analysis. This simulation workflow is shown in Figure 1.

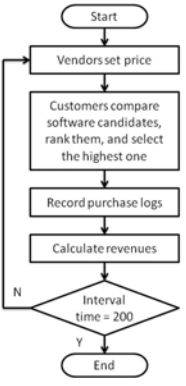


Fig. 1. Simulation Workflow

3.2 Customer Agents and Decision Making

Customer agent parameters are populated according to the size of the organization that the agent represents: small, medium, and large. Each organization has different preferences and criteria based on its size. These criteria have been identified in previous literature about software selection and SaaS adoption. The tasks of customer agents are to compare software products, rank them, and then make a purchase decision for one of the software options.

In this simulation, the customer agent decision support mechanism is based on the Analytic Hierarchy Process (AHP) technique. First, each customer agent has to perform pair-comparisons, in order to create its weight for each criterion [13]. Then, they rate the criteria based on their IT needs. The assumptions on customer needs are based on software selection decision making literatures [1, 3-9]. In our simulation, these ratings only differ because of the size of the organization. The decision hierarchy is illustrated in Figure 2.

The four fundamental groups of criteria in the software selection decision making are: *Finance, Software Capability, Organization,* and *Vendor*.

Finance is usually the top priority for chief information officers (CIOs) of companies. The Finance criterion includes *Benefit* and *Cost*. The Benefit criterion refers to the value that buyers gain from using the software. It can be classified into *Direct Benefits* and *Indirect Benefits*. Direct Benefits refer to the utility that a customer gains from using the software. The customer utility function is defined as:

$$U = (T_u \times N_u \times N_f \times V_f \times u) - TC \quad (1)$$

where T_u is the software usage period (in months), N_u is the number of users, N_f is the number of software functions, V_f is the average value of all software functions, u is the utilization of software. The values for the utilization are usually set between 0.6% and 0.8% for perpetual software [2]. *TC* denotes the total cost of the software.

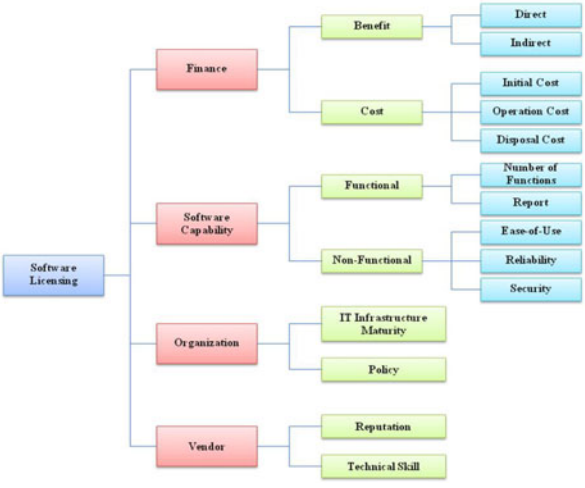


Fig. 2. Software Licensing Decision Hierarchy

The Indirect Benefit gained from using SaaS includes the increase in public image of the organization and the increase in knowhow within the organization. The Indirect Benefit for perpetual software comes from the authority in data management, and the ownership of hardware and software licenses.

The total cost of software *TC* includes *Initial Cost* (or capital cost), *Operation Cost*, and *Disposal Cost*. Initial Cost refers to a one-time payment to a vendor. Operation Cost refers to cost that users have to burden during software usage period. Disposal Cost describes the expenditure, if users abandon the use of the software application. The Operation Costs for perpetual software and for SaaS are defined as:

$$C_{OP\,(PS)} = T_u \times C_{server} \tag{2}$$

$$C_{OP\,(SaaS)} = T_u \times N_u \times (C_n + C_{sub}) \tag{3}$$

where C_{server} is the operation cost for operating a server running perpetual software. C_n is the network transfer cost, and C_{sub} is the subscription rate or pay-per-use cost.

The Software Capability criteria refer to software performance. It includes *Functional* and *Non-Functional* factors. Functional factors refer to software attributes that are related to software output, which can be classified by the *Number of Functions* and the *Number of Reports*. Non-Functional factors include *Ease-of-Use*, *Reliability*, and *Security*.

The Organization criteria refer to factors within an organization that affects new software implementations. These criteria include *IT Infrastructure Maturity* and *Policy*. IT Infrastructure generally includes staff, hardware equipment, space for clients and computers, firm culture, and network infrastructure. The Policy factor describes the organization's policy towards IT.

The Vendor criteria include *Reputation* and *Technical Skill*. Vendor reputation can be indicated by the number of users or clients [6]. The Technical Skill factor includes user training, technical support, experience of developing and using IT products and IT services.

In this paper, by using these factors, customer agents score each software option, and then select the highest score software option. The score depends on factors (e.g., price, vendor reputation, and technical skill) that are set by vendor agents. The chosen software product (i.e., either SaaS or perpetual software) is the most suitable product according to the customer agent profile.

3.3 Dynamic Pricing Schemes

3.3.1 Derivative-Follower (DF) Pricing

This pricing scheme is suited for vendors, who have little knowledge about market conditions [16]. Using this algorithm, at time interval t, a vendor charges a customer a price p_t. The vendor agents incrementally increase (decrease) the price until the profit falls. Then, the agents lower (increase) the prices until the profit falls again. Therefore, price p_{t+1} at time $t + 1$ is set according to:

$$p_{t+1} = p_t \pm \delta \tag{4}$$

where δ is a constant. This process repeats infinitely.

In our simulation, the default value for δ_{SaaS} is 1, while δ_{PS} is 10. The SaaS price is estimated to be in the range [41, 150] and, for PS, in the range [410, 1200]. These values are based on data from CRM software products available in the market. The value of δ is added or subtracted, depending on which pricing direction a vendor is adopting. δ is added if the vendor increases the price.

3.3.2 Demand-Driven (DD) Pricing

Following this pricing scheme, a vendor agent maximizes profit by using knowledge of customers and competitors. The vendor requires perfect information on customer preferences and competitor prices in order to calculate the price. This pricing is the most computing intensive algorithm in our simulation. At time interval $t+1$, vendor charges the customer the price:

$$p_{t+1} = \arg \max_{pM} D(Customer, Competitor, p_M) \tag{5}$$

where $D(Customer, Competitor, p_M)$ denotes the decision support mechanism, which requires knowledge about customer preferences and competitors. We implement the AHP technique as described in Section 3.2 as the decision support mechanism. To obtain the maximum result, the vendor runs the mechanism with prices p_M between $p_t - \alpha$ and $p_t + \alpha$, where α is constant. In our simulation, α_{SaaS} is set to 10 and α_{PS} is set to 100. Then, the vendor sets the price p_{t+1} equal to the price p_M that gave the maximum score.

3.3.3 Competitor-Oriented (CO) Pricing

In the software industry, obtaining market share is the main objective for software vendors, especially if there is only a slight product/service differentiation [10]. This algorithm is adapted from that market competition scheme. At first, the vendor agent

needs to select the competitor to compete with. Then, the vendor simply decreases the price just below of the rival's price. This algorithm requires perfect information of the rival's price. At time interval $t+1$, the vendor sets the price:

$$p_{t+1} = p_{C(t)} - \beta \tag{6}$$

where $p_{C(t)}$ denotes to rival's price at time t. β is a constant, although parameter β could be calculated using the algorithm described in Section 3.3.2. For our simulation, β_{SaaS} is fixed to be 10 and β_{PS} is chosen to be 200. The parameter β will always be subtracted, since a vendor needs to sell at a cheaper price than its rival to gain market share.

4 Implementation and Validation

We developed the simulation model and its agents using Microsoft Excel 2007 due to its rapid development and powerful spreadsheet features. It is suitable for "what if" analyses in the initial state of research. The scripts are written in Visual Basic.

The population of customer agents is 3. Each of the customer agents represents a small, medium, or large firm, respectively. The number of users denotes the total number of employees within the organization. The values used are based on the European Commission's SME guide. Software usage times are based on the assumption that smaller enterprises have less data to process. The values of the customer agent parameters are shown in Table 1.

Table 1. Customer Agent Parameters and their Values

Size	Number of Users	Usage Times
Small	20	24 months
Medium	100	36 months
Large	250	60 months

The environment simulation worksheet controls all actions. The simulation run time of all simulations is 200 time intervals. There are four main vendor agents in this simulation. Each vendor agent represents a popular SaaS vendor (V_1), a popular PS vendor (V_2), an unpopular SaaS vendor (V_3), or an unpopular PS vendor (V_4). The SaaS prices of popular vendors are based on CRM products of Salesforce's CRM Professional Edition. The PS prices of popular vendors are based on Microsoft's CRM on-premise software. The unpopular product prices are set a little bit lower than the prices of the popular products, as illustrated in Table 2. The reputation and skill parameters of unpopular vendors are set lower than those of popular vendors. Since it is hard to obtain actual values for software cost factors (i.e., development cost, maintenance cost, and support cost) from SaaS and PS vendors, we roughly approximate them as $40 for SaaS and $400 for PS.

Table 2. Vendor Agent Parameters and their Values

Vendor	Category	Initial Price	Subscription Rate	Reputation	Skill	Software Cost
V_1	SaaS	0	$65	10	10	$40
V_2	PS	$800	-	10	10	$400
V_3	SaaS	0	$55	7	7	$40
V_4	PS	$700	-	7	7	$400

The AHP-based decision support model is implemented through worksheets, comprising the pairwise comparison worksheet, the software candidate scoring worksheet, and the weight ranking worksheet. Customer agents score the different software options using the respective worksheet. Quantitative criteria, which can be calculated (e.g., cost and customer utility), are scored based on actual values. Qualitative criteria that cannot be calculated are scored based on the agent's properties (e.g., firms' IT infrastructure maturity level and vendor reputation), using values between 1 and 10.

4.1 Price Sensitivity Analysis of Customer Agents' Decision Making

In this section, we perform an analysis of how sensitive customer agents are to prices with respect to their decision making. We run simulations with different values of the price parameter, ranging from 1 to 1500 and calculate the weights, with which the different software options are scored. The results of the price sensitivity analysis are shown for small, medium, and large firms in the following three figures.

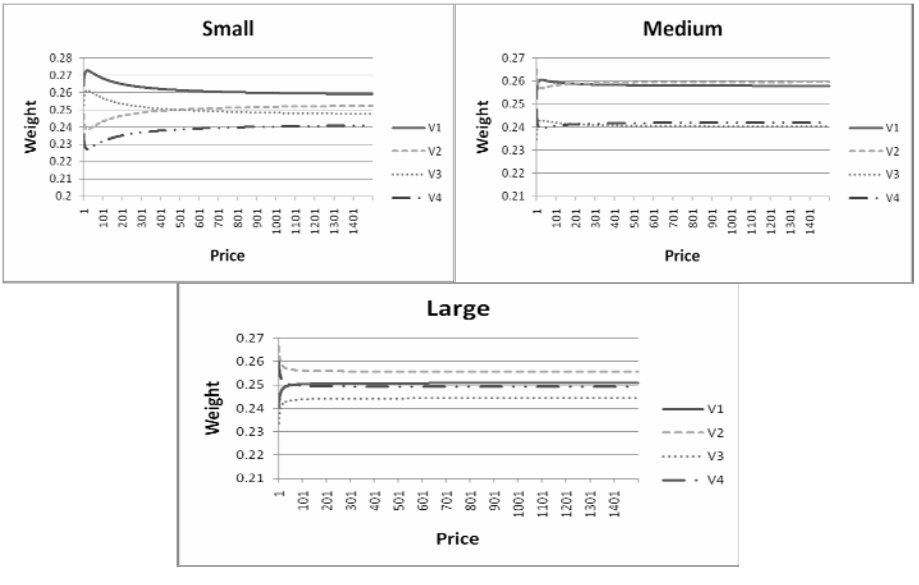


Fig. 3. Results of the Price Sensitivity Analysis for Small, Medium, and Large Firm Agents

In Figure 3, for small firms, as long as the software price is below $488, the most appropriate software option is V_1 (software of the popular SaaS vendor). The options V_3 (software of unpopular SaaS vendor), V_2 (software of popular PS vendor), and V_4 (software of unpopular PS vendor) are scored lower. If the price is higher than $488, V_2 becomes slightly the second best software option instead of V_3. We can imply from these results of our decision making model that SaaS is appropriate for small firms, even if the SaaS option is provided by an unpopular vendor.

In case of medium firms, V_1 is the best option as long as the price is lower than $200. The order of all software options is V_1, V_2, V_3, and V_4. If the price increases further, V_2 becomes the best software option. The overall order is changed to V_2, V_1, V_4, and V_3, where V_2 and V_1 as well as V_4 and V_3 have very similar scores. These results imply that medium companies are indifferent to PS or SaaS products, if the price is larger than $200. Compared with small firms, medium firms already prefer PS options at a lower price, i.e., at $200 instead of at $488. This is caused by the increased usage of the software.

For large firms, apart for very low prices, the selection order is V_2, V_1, V_4, and V_3. The difference in scores between V_2 and V_1 as well as between V_4 and V_3 is significant. This result shows that a large company favors PS software. The score of the PS option of an unpopular vendor is even very close to the score of the SaaS option of a popular software vendor.

4.2 Derivative-Follower (DF) Pricing Simulations

We compare the DF pricing scheme with other pricing schemes, while still differentiating between popular and unpopular vendors. The software options (V_1 and V_2) of the popular SaaS and PS vendors using DF compete with software options (V_3 and V_4) of unpopular SaaS and PS vendors, who use all other pricing schemes. The left column of Figure 4 illustrates how vendors V_1, V_2, V_3 and V_4 set the prices using different pricing schemes. The right column represents how vendors' pricing affect their sales. In this simulation, market share is represented by vendors' sales volume.

If all vendors implement DF (Figure 4, DF vs. DF), SaaS vendor V_1 and PS vendor V_2 gain competitive advantage and increase their prices in the beginning. Actually, V_1 and V_2 are the only vendors that achieve revenue during this time. SaaS vendor V_3 and PS vendor V_4, who do not generate revenue, counter with reducing their prices. Around time $t = 20$, V_3 starts making revenue and increases its price. V_3 slowly overtakes V_2 in revenue at around time $t = 100$. PS vendor V_2 continues raising its price until reaching the maximum price $P_2 = \$1{,}180$, which means, if V_2 charges a price higher than $1,180, customers would switch to SaaS or substitute PS products such as V_4. PS vendor V_4 cannot make any revenue until time $t = 55$ when V_2 charges at maximum price and customers switch to cheaper products. As shown in Figure 4, DF vs. DF, right panel, SaaS vendors with DF (V_1 and V_3) successfully gain a higher market share than PS vendors, who implement DF.

In case of the comparison between DF and DD, V_1 and V_2, who are offered by popular vendors, show an advantage over V_3 and V_4 in the beginning (Figure 4, DF vs. DD, right column). Vendors' software options V_1 and V_2 generate revenues through increased prices until time $t = 55$. At that point, the vendors of V_3 and V_4, who set

lower prices, begin making profit and getting market share. Note that, although V_1 and V_2 gain the most revenue in this market place, their maximum revenue is lower than the revenue of DF vendors competing against unpopular DF vendors.

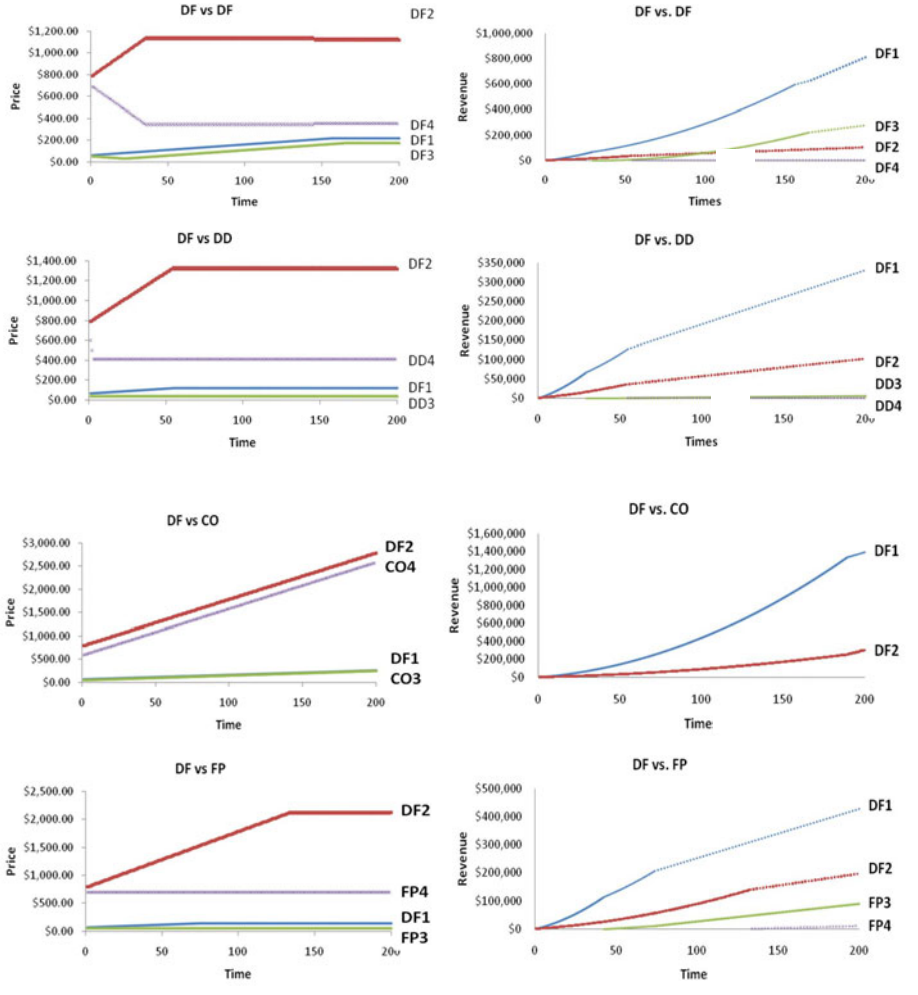


Fig. 4. Price Competition of Popular Vendors Who Use DF with Unpopular Vendors

For the DF vs. CO comparison (Figure 4), the prices of V_1 and V_2 also follow the DF pricing plan, and the prices of V_3 and V_4 are based on the CO pricing scheme. While the price of V_3 follows its rival's price of V_1, the price of V_4 follows the one of V_2. The result shown in the right column illustrates that only the vendors of V_1 and V_2 can gain revenue. Although V_3 and V_4 offer lower prices, the unpopular CO vendors fail to compete successfully with popular DF vendors.

If prices for V_3 and V_4 are fixed in the case of DF vs. FP, the vendors of V_1 and V_2 rapidly gain revenue and even continue to increase prices. This continues until time

$t = 43$, when the V_1 price reaches its maximum. Unlike DF vs. DF, the flat rate SaaS vendor V_3 cannot overcome V_2 vendor's revenue within the simulation duration. The result shows that vendors using DF are more successful than vendors using FP.

From the results shown in Figure 4, we can derive that popular vendors using the DF pricing scheme make the highest revenue against unpopular vendors using CO, DF, FP, or DD, respectively. We can imply further that SaaS vendors are suited for implementing DF against any pricing scheme, while PS vendors are recommended to implement DF pricing against competitors using fixed price or low dynamic pricing.

4.3 Demand-Driven (DD) Pricing Simulations

This simulations compare the pricing and revenues of popular vendors (V_1 and V_2) using DD with those of unpopular vendors (V_3 and V_4), who use DF, DD, CO, or FP pricing schemes.

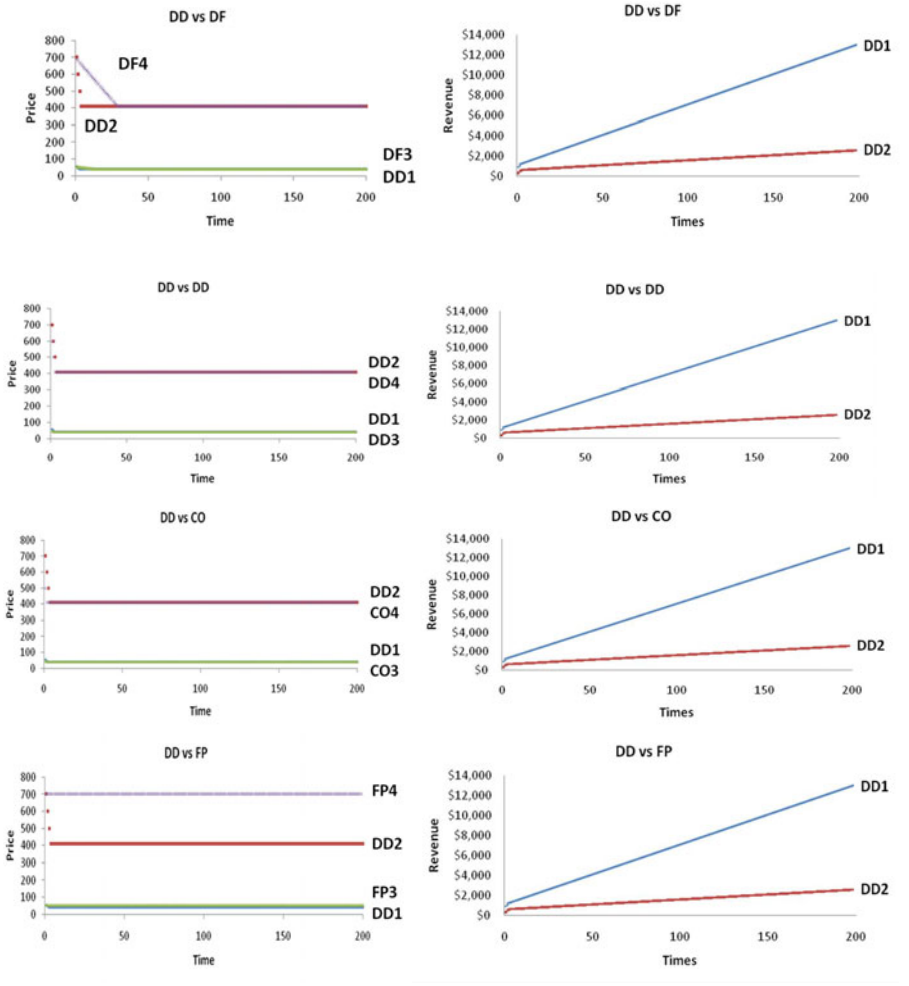


Fig. 5. Price Competition of Popular Vendors Who Use DD with Unpopular Vendors

The results in the right column of Figure 5 show that, if popular vendors implement DD against unpopular vendors with other schemes, the prices and revenue are very similar.

All vendors try reducing prices close to the lowest possible price (i.e., the price that equals cost). After a few simulation intervals, the prices of V_1 and V_2 have already reached the lowest prices. The vendors using DD try maximizing customer utility, regardless of the revenue. The results show that they only receive minimum revenues. As a consequence of the workings of the pricing schemes, V_3 and V_4 are also forced to reduce their price close to the lowest price.

It is also to be noted that only the vendors of V_1 and V_2 are able to make revenues. These revenues gained are low. Although DD vendors successfully capture market share as shown in the right column of Figure 5, the maximum revenue they gained is very low compared with the DF pricing scheme (Figure 4). The maximum revenue of V_1, who implements DD, is about $13,000. With DF pricing (Figure 4), the vendor of V_1 generates the maximum revenue of about $1,400,000, a factor 10 higher. This can be explained by the fact that, beside prices, the popularity of vendors is also important. Therefore, if prices are similar, products with a large popularity can gain a competitive advantage in the market. This is also supported by the findings that we obtain from comparing the graphs of Figure 4, DF vs. DD, with the graph of Figure 5, DD vs. DF. In both graphs, only those pricing schemes gain market share, which are implemented by the popular vendors (i.e., DF in Figure 4 and DD in Figure 5).

From these results, it can be implied that popular SaaS and popular PS vendors should only implement the DD pricing scheme, if they need to gain market share.

4.4 Competitor-Oriented (CO) Pricing Simulations

In this set of simulations, popular vendors (V_1 and V_2) using CO compete with unpopular vendors (V_3 and V_4), who use the other pricing schemes. In the CO pricing simulations, CO vendors try to charge prices lower than their competitors.

The left column of Figure 6 (CO vs. DF, CO vs. DD, and CO vs. CO) shows that the vendors of V_1 and V_2 keep setting their prices lower than their competitors (V_3 and V_4). The vendors of V_3 and V_4 respond by reducing their prices as well. The price competition continues until each vendor reaches the lowest price (i.e., the price that equals cost). The results shown on the right column illustrate that only the vendors of V_1 and V_2 are able to make revenues. This situation is similar to the results of the DD pricing simulations. Each popular vendor, who competes on price, makes low revenues but gains market share. Only in the case of CO vs. FP, the CO pricing does not reach the lowest price and generates revenue that are higher than the one of DD. The reason for this is that vendors using FD do not adapt their prices to the low price of the vendors using CO. Therefore, the CO prices do not reach the minimum price.

Concluding, as in the previous simulations, the popularity of vendors is significant for the success of the pricing plan in obtaining market share. The comparison of the graph DF vs. CO of Figure 4 with CO vs. DF of Figure 6 illustrates this. The same goes for the comparison of the graph DD vs. CO of Figure 5 with the graph CO vs. DD of Figure 6. However, the payoffs are only low revenues.

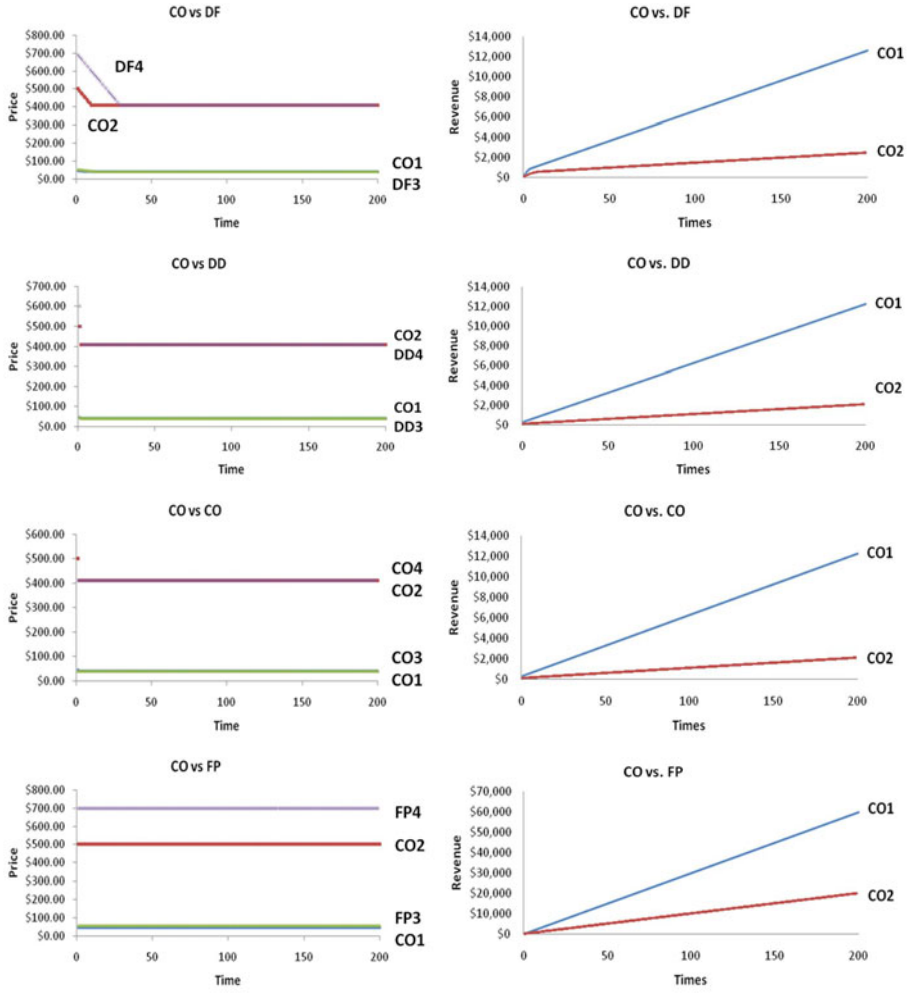


Fig. 6. Price Competition of Popular Vendors Who Use CO with Unpopular Vendors

4.5 Comparison of All Pricing Schemes

In this section, we perform all pricing scheme simulations at the same time. The number of vendor agents is increased to 16, offering 16 different software options ($V_1 - V_{16}$) to the firm agents. The details about the pricing scheme and reputation of vendor agents are shown in Table 3.

Figure 7 illustrates the result of pricing scheme simulations. The outcome shows that only 4 software vendors (V_5, V_6, V_9, and V_{10}) can successfully attract customers. All of them are popular vendors. The most successful vendor is V_9. It is the popular SaaS vendor, who uses DD. The succeeding ones are V_5 (popular SaaS vendor using DF), V_{10} (popular PS vendor using DD), and V_6 (popular PS vendor using DF).

Table 3. Pricing Scheme and Reputation of Vendor Agents

Vendor	Type	Scheme	Reputation	Vendor	Type	Scheme	Reputation
V_1	SaaS	FP	Popular	V_9	SaaS	DD	Popular
V_2	PS	FP	Popular	V_{10}	PS	DD	Popular
V_3	SaaS	FP	Unpopular	V_{11}	SaaS	DD	Unpopular
V_4	PS	FP	Unpopular	V_{12}	PS	DD	Unpopular
V_5	SaaS	DF	Popular	V_{13}	SaaS	CO	Popular
V_6	PS	DF	Popular	V_{14}	PS	CO	Popular
V_7	SaaS	DF	Unpopular	V_{15}	SaaS	CO	Unpopular
V_8	PS	DF	Unpopular	V_{16}	PS	CO	Unpopular

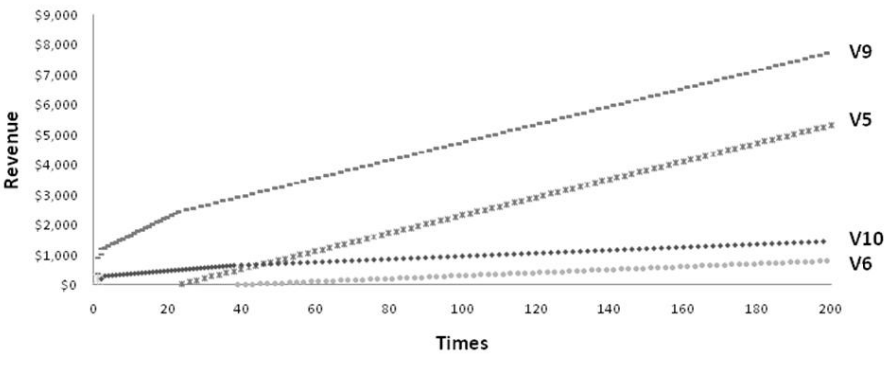


Fig. 7. Comparison of all Pricing Schemes with Respect to Revenue

From the result shown in Figure 7, we conclude that DD is the most effective pricing method in our simulation. Although the DD vendor suffers low profit margin (see low prices in Figure 5), the high sales volume compensates this disadvantage. The DF pricing scheme, which is the second most effective, is proven to be adaptable, due to its ability to gain market share (i.e., revenue) from the other vendors after adapting to the market conditions at time interval 24 (Figure 7).

5 Discussion and Conclusion

In this study, we developed an agent-based simulation of a software market that allows simulating different pricing schemes for two software licensing models: SaaS and PS. The three pricing schemes that have been used are: Derivative-Follower (DF) pricing, Demand-Driven (DD) pricing, and Competitor-Oriented (CO) pricing.

The simulation involves two types of agents: customer agents and vendor agents. The task of customer agents is to score and rank software options by using the Analytic Hierarchy Process (AHP) technique. Vendor agents set prices for their software option according to the pricing scheme deployed. The customer agent parameters are set based on the European Commission's SME guide. There are three categories of customer agents, representing small, medium, or large enterprises. Vendor agent

parameters are based on real world products from popular vendor such as Salesforce.com and Microsoft. Vendor agents are further categorized into SaaS and PS vendors, which, in turn, are divided into popular and unpopular vendors.

Our simulation of two popular vendors and two unpopular vendors show that, for popular SaaS vendors, it is recommended to implement DF against competitors, who adopt dynamic pricing schemes such as DF and CO. It is not suggested to compete with DF against less dynamic pricing schemes such as fixed rate or DD. Popular PS vendors are recommended to adopt DF against any pricing scheme. SaaS and PS vendors should only implement the DD pricing scheme, if they need to gain market share. The CO pricing scheme always offers lower prices than pricing schemes of competitors, i.e., it generally attracts customers and, therefore, helps also gaining market share. However, if the competitors' price is already low, the revenue gained from CO pricing might be too low for running a sustainable business.

If simulating at the same time several popular vendors using different pricing schemes, the simulation results show that the software vendor using DD gained the highest revenue, while the software vendor using DF achieved the second highest only. However, in a real market, obtaining perfect information about customers and competitors is almost impossible, making the DD pricing scheme difficult to execute. Nonetheless, the decision making model introduced in Section 2 could serve as the guideline for implementing the DD pricing scheme in a real world setting.

In this paper, the agents are still not adaptive. Therefore, we plan as part of our future work to integrate learning algorithms. Besides, we plan to allow agents to respond to competitors' actions asynchronously. Finally, we will further study how to reach equilibrium outcomes.

References

1. Wailgum, T.: What to Ask Before Saying Yes to SaaS, Cloud Computing. In: The New York Times, October 27 (2008)
2. Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., Zaharia, M.: Above the Clouds: A Berkeley View of Cloud Computing. Reliable Adaptive Distributed Systems Laboratory, UC Berkeley (2009)
3. Kaplan, J.: SaaS Survey Reveals Real World Experience, http://itmanagement.earthweb.com/features/article.php/3873121/SaaS-Survey-Reveals-Real-World-Experience.htm
4. Colombo, E., Francalanci, C.: Selecting CRM Packages Based on Architectural, Functional, and Cost Requirements: Empirical Validation of a Hierarchical Ranking Model. Requirements Engineering 9, 186–203 (2004)
5. Jadhav, A.S., Sonar, R.M.: Evaluating and Selecting Software Packages: A Review. Information and Software Technology 51, 555–563 (2009)
6. Godse, M., Mulik, S.: An Approach for Selecting Software-as-a-Service (SaaS) Product. In: IEEE International Conference on Cloud Computing (2009)
7. Lai, V.S., Trueblood, R.P., Wong, B.K.: Software Selection: A Case Study of the Application of the Analytical Hierarchical Process to the Selection of a Multimedia Authoring System. Information & Management 36, 221–232 (1999)

8. Mulebeke, J.A.W., Zheng, L.: Analytical Network Process for Software Selection in Product Development: A Case Study. Journal of Engineering and Technology Management 23, 337–352 (2006)
9. Rohitratana, J., Altmann, J.: Software Resource Management Considering the Interrelation between Explicit Cost, Energy Consumption, and Implicit Cost: A Decision Support Model for IT Managers. In: Multikonferenz Wirtschaftsinformatik (2010)
10. Lehmann, S., Buxmann, P.: Pricing Strategies of Software Vendors. Business and Information Systems Engineering 6, 452–462 (2009)
11. Marn, M.V., Roegner, E.V., Zawada, C.C.: Pricing New Products. The McKinsey Quarterly, http://www.mckinseyquarterly.com/article_print.aspx?L2=16&L3=19&ar=1329
12. Kephart, J.O., Hanson, J.E., Greenwald, A.R.: Dynamic Pricing by Software Agents. Computer Networks 32, 731–752 (2000)
13. Saaty, T.L.: The Analytic Hierarchy Process. McGraw-Hill, New York (1980)
14. Matsuda, Y., Whang, S.: Dynamic Pricing for Network Service: Equilibrium and Stability. Management Science 45, 857–869 (1999)
15. Dasgupta, P., Hashimoto, Y.: Multi-attribute Dynamic Pricing for Online Markets using Intelligent Agents. In: Third International Joint Conference on Autonomous Agents and Multiagent Systems, New York, pp. 277–284 (2004)
16. Dasgupta, P., Das, R.: Dynamic Pricing with Limited Competitor Information in a Multi-Agent Economy. In: Scheuermann, P., Etzion, O. (eds.) CoopIS 2000. LNCS, vol. 1901, pp. 299–310. Springer, Heidelberg (2000)
17. Altmann, J., Rhodes, L.: Dynamic Netvalue Analyzer - A Pricing Plan Modeling Tool for ISPs Using Actual Network Usage Data. In: IEEE WECWIS2002, International Workshop on Advance Issues of E-Commerce and Web-Based Information Systems, Newport Beach, USA (2002)
18. Altmann, J., Varaiya, P.: INDEX Project: User Support for Buying QoS with regard to User's Preferences. In: IWQOS 1998, Sixth IEEE/IFIP International Workshop on Quality of Service, Napa, USA, pp. 101–104 (1998)
19. Altmann, J., Rupp, B., Varaiya, P.: Internet Demand under Different Pricing Schemes. In: ACM EC 1999, ACM Conference on Electronic Commerce, Denver, Colorado, USA (1999)
20. Risch, M., Altmann, J.: Enabling Open Cloud Markets Through WS-Agreement Extensions. In: Service Level Agreements in Grids Workshop, in conjunction with GRID2009, Banff, Canada. CoreGRID Springer Series (2009)

SLA-Based Management of Software Licenses as Web Service Resources in Distributed Environments

Claudio Cacciari[1], Daniel Mallmann[2], Csilla Zsigri[3], Francesco D'Andria[4], Björn Hagemeier[2], Angela Rumpl[5], Wolfgang Ziegler[5], and Josep Martrat[4]

[1] CINECA, 40033 Casalecchio di Reno, Italy
c.cacciari@cineca.it
[2] Forschungszentrum Jülich GmbH, 52428 Jülich, Germany
{d.mallmann,b.hagemeier}@fz-juelich.de
[3] The 451 Group, London, WC1E6HH, United Kingdom
csilla.zsigri@the451group.com
[4] Atos Origin - Research and Innovation, Barcelona, 08011, Spain
{francesco.dandria,josep.martrat}@atosorigin.com
[5] Fraunhofer Institute SCAI, Department of Bioinformatics, 53754 Sankt Augustin, Germany
{wolfgang.ziegler,angela.rumpl}@scai.fraunhofer.de

Abstract. Until recently the use of applications requiring a software license for execution was quite limited in distributed environments. Due to the mandatory centralised control of license usage at application runtime, e.g. heartbeat control by the license server running at the home site of a user, traditional software licensing practices are not suitable especially when the distributed environment stretches across administrative domains. In this paper we present a novel approach for managing software licenses as web service resources in distributed service oriented environments. Licenses become mobile objects, which may move to the environment where required to authorise the execution of a license protected application. A first implementation has been realised for dynamic Grid environments in the European SmartLM project co-funded by the European Commission. The SmartLM solution decouples authorisation for license usage from authorisation for application execution. All authorisations are expressed as and guaranteed by Service Level Agreements. We will present the core technology, discuss various security aspects and how they are addressed in SmartLM, and present a number of usage scenarios leveraged by the SmartLM technology. Finally, we will give an outlook on specific issues and current work extending the solution to Clouds and service based systems in general.

1 Introduction

So far, commercial software is rarely used in Grids due to the limitations both with respect to the license management technology and the missing business models of the independent software vendors (ISV) for using their software in the

J. Altmann and O.F. Rana (Eds.): GECON 2010, LNCS 6296, pp. 78–92, 2010.

Grid. Only recently MathWorks has provided a technical solution (and a business model) allowing to use their MATLAB suite in the EGEE Grid environment [14]. However, this is a bilateral agreement only and has so far no implications for using MathWorks software in other Grids like the German D-Grid.

The license management technology for software licenses is still based on the model of local computing centres providing both resources for computation and the software used for simulations together with the required licenses locally. Thus, these licenses are provided on the basis of named users, hostnames (IP-addresses), or sometimes as a site license for the administrative domain of an organization. If we want to use this software in a distributed service oriented infrastructure, using resources that are spread across different administrative domains, that do not host the application's license server, we run into trouble. The licenses usually are bound to hardware within the domain of the user and do not allow access from outside, e.g. due to firewalls, thus, enforcing local use of the protected applications only. In contrast, Grid environments are usually spread across multiple organisations and administrative domains. Even worse, extending Grids to virtualised infrastructures, like utility and cloud computing, introduces additional limitations since the underlying hardware and their performance indicators, e.g. CPU type and frequency are hidden, while they are often used in license agreements.

While current mechanisms almost inhibit the use of licensed software in Grid environments and virtualised infrastructures, the increasing role these environments play in resource provisioning requires a solution. Traditional licensing practices are under pressure from a variety of alternative options (Software as a Service, open source, low-cost development environments, and the increasing software piracy [15] etc.) and are tightening vendors profit margins, pushing down licensing costs and giving more negotiating power to users. Licensors may expect licensees to buy additional licenses for each processor that executes the licensed software (multiplied software fees). This is definitely not viable in a Grid or Cloud context because it limits their flexibility and elasticity. On the one hand, software manufacturers need to change the way licensing works and use flexible and non-hardware based licensing solutions that better fit into a virtual environment (one of the top ten obstacles for Cloud Computing mentioned in citecloud). On the other hand, Software as a Service is going mainstream as final users are asked for paying only for what they use. End users want fairness and flexibility and software vendors do not vote for a reduction in revenue. Hence, the achievement of a win-win situation between software vendors and software users is the main requirement for a mutually advantageous change. The license management has to provide the technological basis for this. A license management product that overcomes current limitations and provides manifold benefits for all parties involved makes a compelling selling proposition here. The SmartLM project [18] will provide a generic and flexible licensing virtualisation technology based on standards for new service oriented business models.

To our best knowledge little research has been focusing on licensing technology since the new IT infrastructure paradigms Grids, Clouds and SOA became

serious enhancements of traditional IT infrastructures. Early approaches like [3], [6] and [8] propose front-ends to the FlexNet License manager [5] providing scheduling and reservation of licenses. However, both approaches assume open firewall ports at runtime to enable the communication between license manager and application. [4] focusses on maximisation of license usage and resource usage in Grids. Like the previously mentioned approaches open firewall ports at runtime are a prerequisite. Other approaches like [11] and [13] stem from the P2P environment. The former addressing licensing of music sharing while the second one is more generally addressing content sharing. However, both approaches grant unlimited access or usage once a license has been issued and thus do not support a business model useful for ISVs. [10] finally is proposing a license mechanism suitable for SOA environments. However, the paper sketches the architecture and some possible interactions but lacks an implementation and experiments with real applications. Moreover, the approach also assumes open firewall ports at runtime. Only recently when these new paradigms gained ground in productive environments where e.g. more commercial simulation codes are used than in the e-Science domain license technology came to the fore. In [12] the authors give an overview on current licensing technology and models and describe two approaches developed in European projects to overcome the limitations. One of the presented approaches breaks with the current technology and is currently implemented in the SmartLM project while the second approach circumvents some of the limitations imposed by the de-facto standard. The authors of [2] describe another approach, which is similar to SmartLM but lacks the integrated accounting and billing service and still requires network connectivity with the ISV when a user requests a license from the local license management server. As we will explain in the following sections SmartLM has taken an holistic approach for all services around license management while effectively making dispensable the requirement for permanent network connectivity between the license management service, the execution site of the application during runtime or to the ISV when requesting a license.In the European project BEinGRID another approach was developed which allows the use of existing licenses in Grid environments through tunnelling of the communication of the license server to the application [16]. While technically feasible this approach rises a number of legal issues since many license contracts limit the use of a software license outside a company or outside a certain radius from the company.

The remainder of the paper is organised as follows. First, we give an overview of the project and of the underlying business models. In Section 3 the architecture is presented. Section 4 discusses security aspects and Section 5 Finally, the paper concludes with a summary and plans for future research.

2 SmartLM Overview

The SmartLM approach reflects the changing paradigms of the information technology. The transition from monolithic infrastructures to agile, service-based architectures was at the origin of the SmartLM project. Treating and implementing software licenses as Web Service resources, thus providing platform independent

access to licenses just like other virtualised resources is the core of the SmartLM architecture [9]. Licenses are managed through a license service implemented as a bag of services (see Section 3) and realised as tokens, which deliver the required flexibility and mobility. Decoupling authorisation for license usage from authorisation for application execution finally allows executing the license protected application without requiring a permanent bidirectional network connection to the license server at runtime controlling the authorisation, like it is the case today. All authorisations for license usage are expressed and guaranteed by Service Level Agreements based on the WS-Agreement specification. Moreover, this solution allows license aggregation: an application service provider can provide the customer with access to applications without having to buy additional licenses. Customers licenses may be combined with the ones owned by the service provider for running complex jobs with different applications. The license mechanism is designed for distributed environments and works with loosely coupled systems, realising software licenses as Web service resources renders licenses adaptable and mobile.

A built-in license scheduling service allows licenses to be reserved in advance. With advance reservation a license can be guaranteed to be available at a later point in time, e.g. when the computing resources to execute the simulation become available. Thus, no risk of blocked licenses while an application is idling waiting for computing resources to become available. Similar, no risk of aborted applications because the required license is used by another user at the time the application starts up after waiting for resources. An orchestration service can be used to synchronise license reservation and resource availability.

Using open standards as far as possible instead of proprietary protocols is considered crucial for the interoperability with and integration into existing middleware stacks. As mentioned before, in SmartLM license usage is governed by Service Level Agreements based on the WS-Agreement specification. A new term language has been defined in the project allowing to express license properties as service terms. Furthermore, in collaboration with the GRAAP working group [7] of the Open Grid Forum (OGF) the WS-Agreement specification [1] has been extended to allow negotiation and re-negotiation (see Section 3.2).

SmartLM has identified three different actors: the user (and his home organisation, which is the licensee), the independent software vendor (ISV), which is the licensor and the application service provider (ASP), which provides the hardware and hosts the applications.

As mentioned before, SmartLM addresses the licensing management issues not only from the technological point of view, but also develops new business models. The reason being that to convince Independent Software Vendors adopting to the new license technology a business perspective must be identifiable.

2.1 New Business Models

Through close collaboration with stakeholders - software vendors, application service providers, end users - we have identified major licensing shortcomings

and have tried to fill in the gaps. The usage-based models presented in the following paragraphs look at different aspects of licensing problems.

1. In the first model we have pinpointed the **Application Service Provider** (ASP) and analysed five cases - customer license housing, embedded license (Dependant Software Vendor, DSV), license redirection (external consultant), license aggregation and license reselling - that companies may most frequently encounter in real operations. In all of them, the ASP plays a central role, being a reseller of hardware, software and services. The introduction of the ASP can be very advantageous for both the ISVs and the end users. From the **ISVs** point of view, the ASPs can generate additional business offering licenses and hardware resources for end users on-demand (competitive resource provision). Making use of economies of scale and SmartLM features, the ASP can make existing models (e.g. short-term licenses) more attractive to customers, and introduce new ones that the ISV is not willing to offer, such as pay-per-use for license reselling.
2. The license extension model allows **end users** to extend their licenses in their Local Area Network (LAN) and distributed environments on-demand, e.g. for workload peaks. The license server takes care and simplifies the process of the extension of licenses, e.g. in terms of accounting and license administration. These mechanisms give end users more flexibility and value and at the same time generate additional revenue for ISVs and ASPs.
3. Currently, most contracts between ISVs and end users restrict the license usage to LAN. The license aggregation model allows the use of licenses that belong to different sites and brings them together to form a single license token. These licenses can come from either the ISV or the ASP. **End users** gain more flexibility and value and get access to huge hardware resources. The ASP provides these hardware resources to the end user and generates additional business for the **ISV**.

3 SmartLM Architecture

The core part of the SmartLM software is the license service. Next to it there are the accounting and billing service, the orchestration service and the Unicore Virtual Organization System (UVOS) [19]. These last two components have been developed in different projects, but are integrated in the SmartLM architecture, as shown in the Figure 1 where the components developed in SmartLM are the dark blocks.

3.1 License Service

The license service is capable to manage all licenses that support the new licensing mechanism owned by a company. It provides a single point for license management and an easy and comfortable access to the entire license information. For redundancy multiple synchronised instances of the service may be

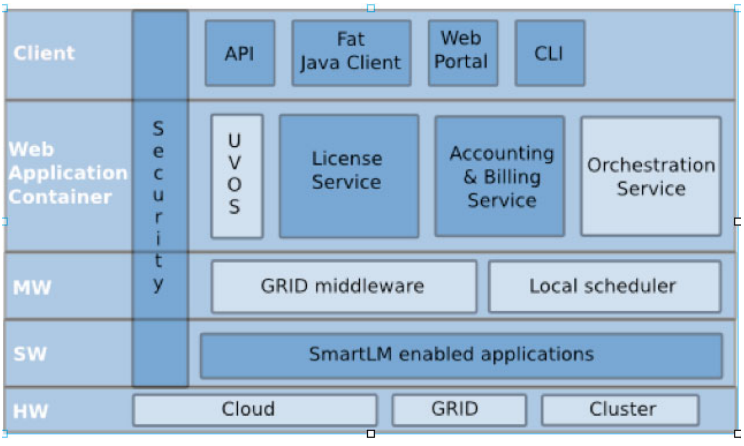


Fig. 1. SmartLM Components

hosted on different servers. Figure 2 depicts the components of the license service and their interaction. When a company buys a license from a software vendor, the license is added to the license service using the *license administration service*. The *license management service* may then fork license tokens from this license once the user's request for a license has been accepted. Policies of the ISV and local policies are used by the *policy management service* to authorise the user's request. A user can request a license by means of one of the different clients available, such as the java based eclipse client, a web portal or the command line interface. The authentication process always relies on the common UVOS service, allowing to store user attributes and validate the user credentials. When a user requests a license for an application or a feature of an application from the license service, the terms of license usage are negotiated between the user and the service through the *SLA and negotiation service* and then embedded in a Service Level Agreement document following the WS-Agreement specification. The negotiation is based on templates specific to the application. The cost resulting from license request is calculated beforehand and becomes part of the SLA. This allows to easily check the license cost against budget constraints for users or user groups implement in UVOS. After creating the token the information relevant for accounting and billing is passed to the accounting and billing service through the *usage record service* in form of a usage record. The *license information service* allows both users and administrators to retrieve information on installed ISV licenses, licenses available or in use. All persistent data of the license service are stored by the *storage service*.

In case of using remote resources the token will be forwarded to the site where the application will be executed either by the user, the *orchestrator* or

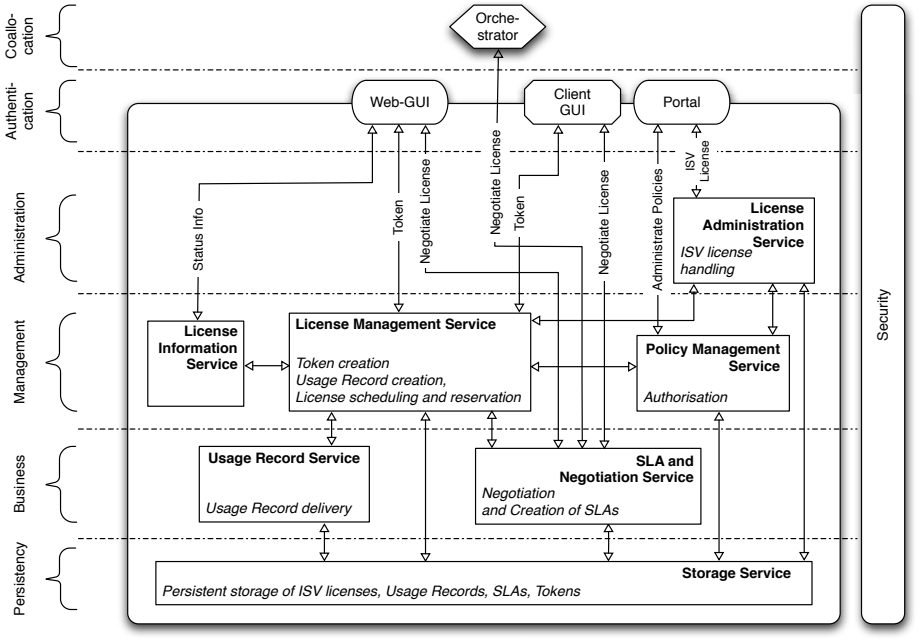


Fig. 2. Architecture of the License Service

the application. To protect against changes the token is signed by the license service, it may also be encrypted to prevent from unauthorised access when traveling towards the computational resource.

3.2 Service Level Agreement and Negotiation

The license service implements the WS-Agreement specification of the OGF for creating and negotiating Service Level Agreements (SLA). WS-Agreement specifies a language and a protocol for advertising the capabilities of service providers, for creating agreements based on templates, and for monitoring agreement compliance at runtime. Like an agreement document, the template is composed of a template name, a context element and agreement terms, but additionally also includes information on agreement creation constraints to describe a range of agreements which are acceptable by the agreement provider. Figure 3 depicts a template with service description terms related to licenses.

SmartLM has adopted and extended the WS-Agreement implementation for Java [20], which is a framework intended to provide an easy way to create, monitor and terminate service level agreements based on the WS-Agreement specification. According to the specification, agreements are created based on specific templates. Therefore, a client queries the agreement templates from a SLA factory service, which exposes them as resource properties. Templates are

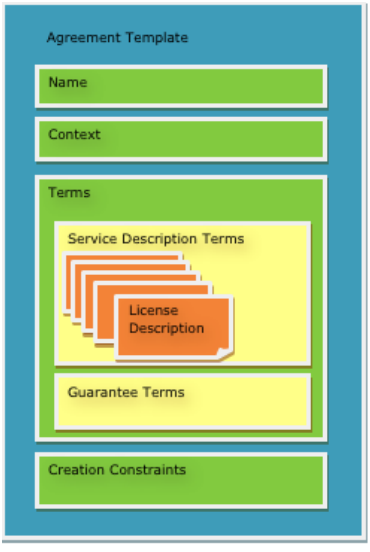


Fig. 3. SLA Template for License Agreements

dynamically created based on the available licenses installed in the license service. Based on a suitable template the client creates a new agreement offer, and may modify the offer. The offer is then sent to the factory service that will create a new agreement and return an Endpoint Reference (EPR) to it. The agreement framework provides also the capability of negotiating templates, which is an improvement of the initial WS-Agreement protocol. When the client changes the retrieved template, it may create a negotiation quote with the modified template and send it to the agreement factory service, which will try to find a suitable template based on the quote. If a template can be provided, the agreement factory service just returns it to the client, indicating that an offer based on that template will potentially be accepted, otherwise it employs some strategy to create reasonable counter offers.

3.3 Accounting and Billing Service

The architecture of the accounting and billing service is presented in Figure 4. It consists of three main blocks: the database to store the usage records, the usage record processing logic for processing the usage records and the rule engine to determine the price of a license. Finally, the accounting and billing portal is used by different SmartLM users to access to the full accounting and billing information generated by licensed applications. In addition, it provides functionalities like budget control triggers of the licensed software, and to define different business rules used to define different pricing policies.

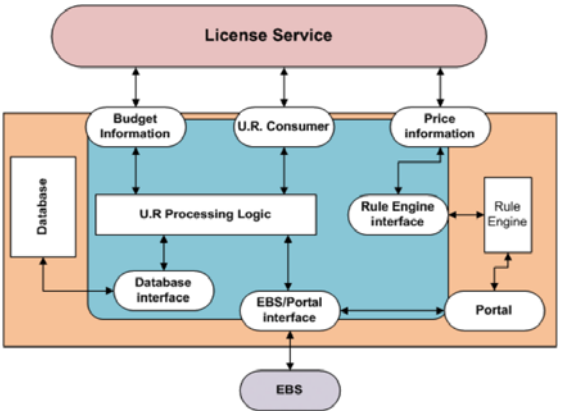


Fig. 4. Architecture of the Accounting and Billing Service

4 Security

All the positive aspects considered so far about the flexibility and the reliability of the SmartLM services would be useless without an adequate level of security. The security issues can be examined from two points of view: the services and the licenses.

4.1 Services

Service security in SmartLM is not much different from service security in other distributed, service based, systems. In the same way as the most important web service engines like Axis2, CXF, etc., SmartLM employs a handler scheme, where incoming and outgoing calls pass through handlers, which deal with authentication and authorisation in a generic fashion. For incoming calls that means that only authenticated and authorised calls will even make it to the service implementation itself. Unauthenticated or unauthorised calls will be rejected already by the incoming security handlers. The project follows this scheme and implements SmartLM specific handlers for authentication and authorisation. Figure 5 shows a generic sequence diagram of a SmartLM component being accessed and doing authentication and authorisation. A client - a user or another SmartLM service - triggers an action in a SmartLM component. This component needs to first authenticate the client and then determine its permissions, i.e. if it is allowed to trigger the requested action. Therefore, it will first call the authentication service (UVOS) to query information about the client. With this information, it will query the policy engine (XACMLight), whether the client is authorised for this action or not. The policy engine will return an appropriate reply, permitting

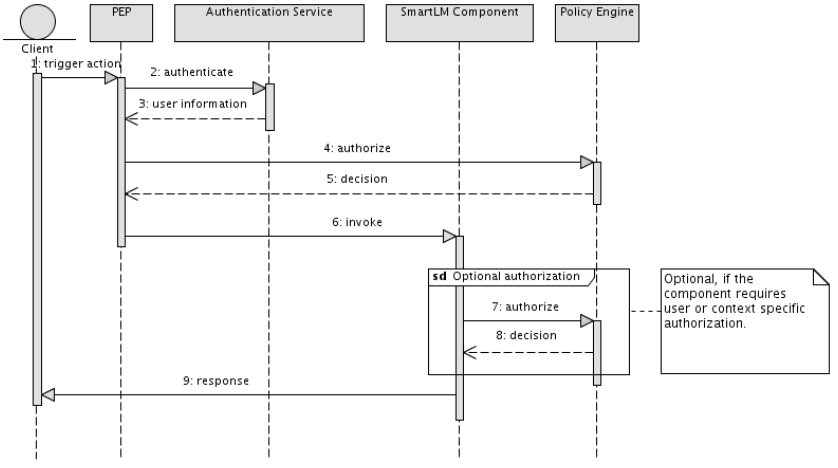


Fig. 5. Authentication and Authorisation Sequence Diagram

or denying access. The use of UVOS allows us to use a single user data base for both middleware and the license server wherever these are located in the same administrative domain.

Policy enforcement is done, before any request enters the service. This allows the use of generic authentication and authorisation components, which can be enabled or disabled for services as needed. Besides authentication and authorisation do not have to be hard wired into the services themselves. As you can also see from Figure 5, we allow for an optional authorisation step inside the service. While this is not generally necessary, as most of the authorisation can be done generically, we foresee occasional need for service specific authorisation, which will be allowed through this. This is true in cases the call to a service interface doesn't allow proper distinction of what resource is affected by the actual call. The service itself may however be able to make this distinction.

4.2 Licenses

Licenses and license tokens are the primary goods in SmartLM. They need to be protected against unlawful use by all means. We expect that all parties involved in the SmartLM license scenario have X.509 certificates [12], which enables them to sign and verify the information transported in a certificate. This expectation can not be satisfied under all circumstances. Most of the time it will be sufficient that only ISVs and license servers at ASPs have full licenses. They are required for the verification of the chain of trust from the ISV to the license server. User certificates are only really needed in Grid scenarios, where the job submission needs to be signed, such that the user cannot refute to have sent the job when billed for its execution. When a service provider buys a license from the ISV, that license will be issued exactly to his license server. The license server identity will

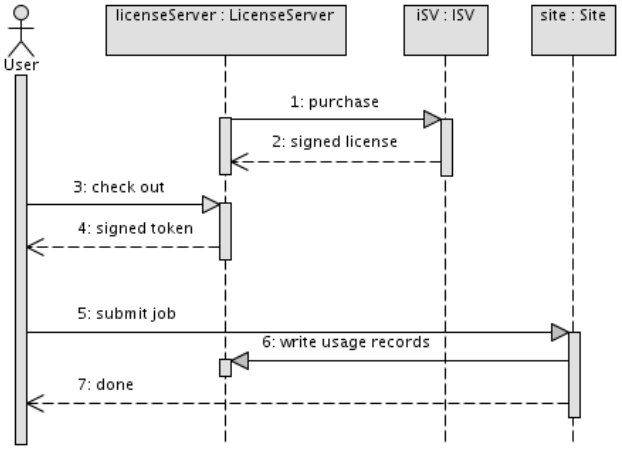


Fig. 6. Purchase and Checkout Sequence Diagram

be part of the license. The entire information will be signed, so it is not possible to copy the license to another license server without also copying the server's identity, i.e. private key, to the other server. Optionally, in order to avoid the lodging of license file in other servers, certain properties of the license server's hardware can be included in the license servers identity. Checking out a license token is similar to buying a license from an ISV (see Figure 6). Again, the token will be signed including information about the receiver of the token, e.g. the job. In order to protect the token against unlawful use, hashes of the jobs input, which have been provided by the user on submission, are contained in the token. Thus, the token is only useful for a particular execution of the software. Different input would require a new token. The license server has to ensure that the sum of the multiplicities of a feature in all license tokens derived from a particular license may never exceed the multiplicity of the respective feature in the parent license. For Example, if a license contains permissions to use a feature on 100 CPUs at any given point in time, then the sum of CPUs for this feature in all the license tokens derived from this license may not exceed this number at any time. This requirement has to be ensured by the license server. This is actually one of the key features where the ISV has to trust the license server to observe this requirement. This trust is expressed by issuing a license to this server and can be verified by the application. The purchase and checkout procedure can be summarised in the following scenario. A company buys a license from an ISV. For this purpose, the company provides its license server's public certificate to the ISV. The ISV then issues the license to the license server of the company. This is ensured by attaching the server's credential information to the license. It can thus be verified that this license is issued specifically to this license server. Only the license server to which a license has been issued can fork license tokens from this license. Information about the original license needs to be conveyed

inside the license token for verification. At the same time, the license token has to contain information about whom it has been issued to and who is allowed to use it. Eventually, the Policy Enforcement Point (PEP) in the application will be able to verify if a correct path of delegations (ISV to License Server to User) has been built and obeyed for the execution. The purchase and checkout procedure is depicted in the following sequence diagram.

5 SmartLM License Usage Scenarios

The SmartLM architecture allows for different usage scenarios of license protected applications. The main distinction of the different scenarios is the availability of a network connection between the SmartLM license server and the protected application at runtime.Within the SmartLM project three ISVs adapted their applications for using the SmartLM API that enables the new license mechanism: with CFX from ANSYS, PERMAS from INTES and OPTIMUS from LMS three heavily used industrial codes are included in the project's use-cases and evaluation. Naturally, replacing the API of a traditional license management system against the SmartLM API is a prerequisite for using the new mechanism.

5.1 Basic Scenario: Without Network Link Available at Run-Time

Figure 7 presents the basic SmartLM license usage scenario covers use cases where no network connection to the SmartLM license server is available during the execution of the protected software. The following two use cases show the range of this scenario:

a) a compute cluster that executes the application is protected through a firewall that does not allow any communication from the cluster to the outside world and vice versa,

b) the application is used on a laptop while traveling without network connection at all.

In both cases it is necessary to

1) negotiate the usage of the license protected software in advance, reserve the license enabling other users using the license before and after the reservation

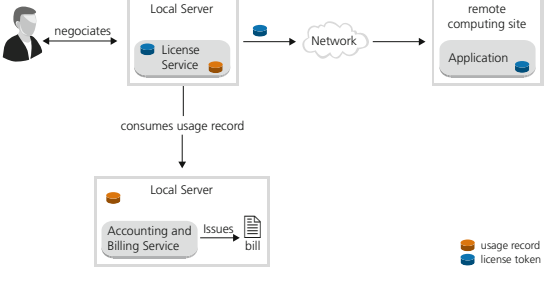


Fig. 7. Basic Scenario without Bi-Directional Network Connection at Run-Time

period while preventing other user from blocking the license during the reservation period. The SLA the user creates when creating the token guarantees the availability of the license while the license scheduler creates the schedule of license usage.

2) receive a license token from the SmartLM license server and

3) transfer the license token to the execution environment before the application is executed.

In a scenario without network connection neither a modification of the license terms at run time-time is not possible, i.e. the user is charged as agreed in the SLA.

5.2 Advanced Scenario: Network Link Available at Run-Time

The advanced SmartLM license usage scenario depicted in Figure 8 covers all use cases where a network connection between the protected application and the SmartLM license server is available during execution, either through a trusted entity or a direct connection to the license server. Two exemplary utilisation characterise the advanced scenario: a) a user submits a job for batch execution to a Grid scheduler that chooses the executing resource by a user defined policy, e.g. optimisation function of time and cost,

b) a user starts an interactive application on a workstation.

In a scenario with a Grid scheduler involved the Grid scheduler will schedule the license token along with the application or the data required for the application execution. For that purpose SmartLM is shipped with an external Orchestrator that provides the Grid scheduler functionality. In the advanced scenario re-negotiation of license terms is possible, i.e. extending or reducing the terms of a license like time or features. This allows to return a license to the license scheduler when the application stops earlier than predicted by the user, e.g. because of a failure. However, local policies may be used to control if tokens can be returned and if there are penalties or rewards defined for this case. The policies will be expressed in the terms of the SLA.

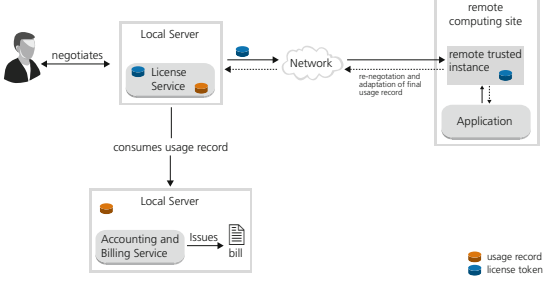


Fig. 8. Advanced Scenario With Bi-Directional Network Connection at Run-Time

5.3 License Aggregation

The token-based licenses leverage new forms of license aggregation for both scenarios described before. A user having access to local licenses is able to aggregate them with licenses owned e.g. by an ASP when using the resources of his ASP to satisfy peak demand. Aggregation is a simple as moving the token generated from the local license to the ASP environment, creating a token from the ASP's license and providing both to the SmartLM API for authorisation. The API then aggregates the license terms from two (or more) tokens and communicates the result to the application.

6 Conclusion and Future Work

We discussed a new approach for software licensing in distributed computing infrastructures based on Service Level Agreements. We presented the basics of the SmartLM technology and implementation, new business models considering the interest of the software vendors, the users and the application service providers. Managing software licenses as Web service resources allows to achieve the flexibility required by distributed environments such as Grid and Clouds, overcoming the limitations of the actual license servers.

The project is currently focussing on three crucial aspects of the licensing process: the re-negotiation of the Service Level Agreements [17], the accounting based on the actual usage of the resources and additional security and trustability of the license tokens. Moreover, additional enhancements like (i) the introduction of trusted clocks limiting the possibilities of cheating by manipulating time and date of the execution environment and (ii) the realisation of a trusted entity are under investigation and will become part of the system in near future.

In parallel we are performing a thorough internal evaluation of functional and non-functional properties of the prototype. The results will be reported in a separate paper in near future.

Acknowledgements

Some of the work reported in this paper has been funded by the European Commissions ICT programme in the FP7 project SmartLM under grant #216759 and in the FP7 project OPTIMIS. This paper also includes work funded by the German Federal Ministry of Education and Research through the D-Grid project under grant #01AK800A. Last not least the authors gracefully acknowledge the stimulating discussions in the GRAAP-WG.

References

1. Andrieux, A., et al.: Web Services Agreement Specification (WS-Agreement). In: Grid Forum Document GFD.107, Open Grid Forum (2007)
2. Dalheimer, M., Pfreundt, F.-J.: GenLM: License Management for Grid and Cloud Computing Environments. In: Proceedings of the CCGrid conference 2009 (2009)

3. Dong, X., Wang, Y., Zheng, F., Guo, H., Yang, S., Wu, W.: Floating license sharing system in grid environment. In: SKG, p. 96. IEEE Computer Society, Los Alamitos (2005)
4. Dong, X., Wang, Y., Zheng, F., Qin, Z., Guo, H., Feng, G.: Key techniques of software sharing for on demand service-oriented computing. In: Chung, Y.-C., Moreira, J.E. (eds.) GPC 2006. LNCS, vol. 3947, pp. 557–566. Springer, Heidelberg (2006)
5. Flexera WebSite, http://www.flexerasoftware.com/products/flexnet-manager.htm
6. Fu, W., Xiao, N., Lu, X.: Sharing software resources with floating license in grid environment. In: NPC 2007: Proceedings of the 2007 IFIP International Conference on Network and Parallel Computing Workshops, Washington, DC, USA, pp. 288–294. IEEE Computer Society, Los Alamitos (2007)
7. Grid Resource Allocation Agreement Protocol Working Group (July 18, 2008), https://forge.gridforum.org/projects/graap-wg/
8. Guofu, F., Yinfeng, W., Hua, G., Xiaoshe, D.: Research on software license manager and sharing system in grid. In: GCCW 2006: Proceedings of the Fifth International Conference on Grid and Cooperative Computing Workshops, Washington, DC, USA, pp. 35–38. IEEE Computer Society Press, Los Alamitos (2006)
9. Hagemeier, B., Mallmann, D., Ziegler, W.: Securing software licenses in a location independent manner. In: Proceedings of Cracow Grid Workshop 2009, pp. 112–119 (2010)
10. Katsaros, G., Antonopoulos, S., Kyriazis, D., Varvarigou, T.: Service Oriented License Providing. In: Proceedings of IEEE International Conference on Service-Oriented Computing and Applications, SOCA (2009)
11. Kwok, S.H., Lui, S.M.: A license management model for peer-to-peer music sharing. International Journal of Information Technology and Decision Making 1(3), 541–558 (2002)
12. Li, J., Wäldrich, O., Ziegler, W.: Towards sla-based software licenses. pp. 139–152 (2008)
13. Liu, Y., Yuan, C., Zhong, Y.: Implementing digital right management in p2p content sharing system. In: Jin, H., Rana, O.F., Pan, Y., Prasanna, V.K. (eds.) ICA3PP 2007. LNCS, vol. 4494, pp. 348–355. Springer, Heidelberg (2007)
14. EGEE now running Matlab parallel computing products (May 25, 2010), http://www.scientific-computing.com/news
15. Seventh Annual BSA/IDC Global software - 09 piracy study (May 25, 2010), http://portal.bsa.org/globalpiracy2009/index.html
16. Raekow, Y., Simmendinger, C., Krämer-Fuhrmann, O.: License management in grid and high performance computing. Computer Science, Research + Development 23(3-4), 275–281 (2009)
17. Rumpl, A., Wäldrich, O., Ziegler, W.: Extending WS-AGREEMENT with multi-round negotiation capability. In: Proceedings of Grid 2009 workshop on SLA usage in Grids). Springer, Heidelberg (2010)
18. SmartLM - Grid-friendly software licensing for location independent application execution (July 18, 2008) http://www.smartlm.eu
19. UVOS - Unicore Virtual Organisation System, February 8 (2010), http://uvos.chemomentum.org/
20. WSAG4J - Web Services Agreement for Java (February 8, 2010), http://packcs-e0.scai.fraunhofer.de/mss-project/wsag4j/index.html

IaaS Adoption Determinants in Enterprises

Christoph Heinle[1] and Jörg Strebel[2]

[1] Karlsruhe Institute of Technology, 76131 Karlsruhe, Germany
ugbnj@stud.uni-karlsruhe.de
[2] Karlsruhe Institute of Technology, Institute of Information Systems and Management (IISM), 76131 Karlsruhe, Germany
strebel@iism.uni-karlsruhe.de

Abstract. Infrastructure-as-a-Service (IaaS) gains increasing importance as a new, flexible provisioning model for IT resources. However, its uptake in the enterprise environment is not without its challenges; among the most important ones are organizational issues. This paper presents an IaaS acceptance model based on multiple theoretical dimensions , which focuses on organizational drivers and barriers to IaaS deployment. It was challenged in a series of expert interviews and the resulting hypotheses give new insights into IaaS adoption drivers and represent a solid foundation for future empirical studies.

Keywords: Infrastructure-as-a-Service, expert interviews.

1 Introduction

1.1 Motivation

Cloud Computing gains importance as a new paradigm of using IT resources of all kind, which are provided 'as-a-service' over the Internet. Many research analysts from e.g. Gartner or Forrester, consider Cloud Computing as one of the most significant trends with a great potential for changing the whole IT industry (e.g. [16]). A number of surveys from IT consulting agencies and news outlets cast a light on the current state of Cloud Computing adoption. The CIO magazine conducted a CIO Cloud Computing Survey in June 2009 with the purpose of measuring enterprise cloud computing adoption among IT decision-makers (c.f. [10]). The participants were CIO.com Website visitors involved in purchasing IT-related products and services. The survey findings are based on 240 responses from IT professionals in a variety of industries including high tech, telecom & utilities, government and nonprofits. Over half of the respondents are the head of IT at their company or business unit. The participating companies evenly varied in size across the whole range (from <$100 million to >$1 billion). The four greatest concerns surrounding Cloud Computing were security, loss of control over data, regulatory/compliance concerns and performance issues. The primary reasons for considering Cloud Computing were stated as reduced hardware infrastructure costs, reduced IT staffing/administration costs, access to skills/capabilities that the enterprise has no interest in developing in-house

J. Altmann and O.F. Rana (Eds.): GECON 2010, LNCS 6296, pp. 93–104, 2010.

and the scalability on demand/flexibility to the business. For almost 68% of the enterprises polled, Cloud Computing is a technology that they are exploring or will explore within the next one to three years. Those findings have been replicated in other market surveys (e.g. by IDC [19]). Hence, the market is embracing this new technology, and enterprises make their first carefully planned steps in the direction of Cloud Computing adoption. However, enterprises also perceive the risks associated with this sourcing approach.

This situation motivates the research question, what the organizational drivers and obstacles of the Cloud computing adoption in enterprises are. Cloud Computing is a rather general concept, so this paper will only focus on Infrastructure-as-a-Service (IaaS) as one of the three main service models in Cloud Computing (unfortunately, there are no IaaS-specific market surveys, to the best of the author's knowledge). The National Institute of Standards and Technology (NIST) gives a concise definition of IaaS: „The capability provided to the consumer is to provision 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 but has control over operating systems, storage, deployed applications, and possibly limited control of select networking components (e.g., host firewalls)."[25]

This paper is structured in five sections. In the following section, an overview of the related literature is given; in this context, the research question is discussed. In Sec. 2, the IaaS adoption model is developed and the research approach is detailed. Sec. 3 features the results of the expert interviews that were used to challenge the adoption model. The results are then discussed in Sec. 4 and compared to the literature references above; an improved adoption model is suggested as a result. The discussion section closes with an analysis of the limitations of this research and a general conclusion.

1.2 Related Work

Cloud computing, let alone IaaS, does not yet feature a developed body of research for the question of enterprise usage determinants. First attempts are found in Xin et al.[32] and Kim et al.[21]; these works are still in their infancy and have to be considered research agendas rather than full-featured research results, especially given the drawback that none of them contains any empirical validation of their theoretical constructs. For the field of Software-as-a-Service (SaaS) adoption, Benlian et al.[5] proposed and tested a multi-theoretical model of SaaS adoption determinants. The results however remained on a very conceptual level; most of the determinants turned out not to influence the SaaS adoption in a statistically significant fashion. Jayatilaka et al. [20] went in a similar direction for Application Service Providing (ASP); they gathered determining factors of ASP choice in an explorative approach.

Grid Computing, as a closely related but older concept, has also attracted a number of researchers investigating the organizational effects of this IT sourcing option.

Hwang et al. [18] identify the most important decision factors of Grid computing solution providers for adopting Open Grid computing technology, i. e. Grid software products for Open Grids. Their research is based on the assumption that an enterprise would want to introduce Open Grid-based solutions, and that this enterprise would then need support for the choice and the implementation of this technology. The identified decision factors seem plausible for the given scenario (e.g. usage of open standards, easiness of implementation), however, they were elicited from Korean solution providers only; the decision factors prevalent in enterprises considering Grid adoption might look different.

Baier [2] bases her research on the technology diffusion model of Hall[17] and distinguishes four dimensions relevant for technology diffusion (benefits, cost, network effects, information and uncertainty). After setting up her technology assessment framework, she uses the application of Grid computing to business information systems as a case study for evaluation. The analysis reveals two factors, observable benefit generation and data security, as the main challenges for Grid computing. This study approaches Grid computing from a microeconomic angle and remains on a highly conceptual level. The six experts who gave their input for the study, all came from an academic background; this set-up limits the external validity of the results.

The paper by Thanos et al.[28] examines the economic factors determining the diffusion of Grid middleware and focuses on the economic forces acting upon it like network effects and market impacts. The research focus lies on the interaction among the Grid participants, each participant (e.g. an enterprise) is seen as a single unit of analysis [27, p. 407]; the decision processes within the participating organizations and their drivers or obstacles were not analyzed. Although this paper discusses some of the factors influencing grid adoption of enterprises, it does so from an overall Grid perspective; the decision process happening on an enterprise level plays a minor role, so its findings are only marginally relevant for this research endeavor.

As a conclusion, the existing literature on Cloud and Grid Computing adoption in the enterprise only shows an incomplete and partly invalid picture of this research topic and does not address the IaaS-specific adoption issues at all. In the Cloud Computing field, only SaaS or ASP-related adoption models have been investigated so far; general Cloud Computing adoption models are in their infancy. In the Grid Computing field, the literature mostly focuses on rather specific research scenarios (Grid markets, solution providers). This paper makes a first attempt to fill this gap by analyzing IaaS-specific adoption determinants in an enterprise setting. The corresponding research question is, what the organizational drivers and obstacles of the IaaS adoption in enterprises are (organizational drivers are those that are related to organizational issues of either the client or the IaaS provider or their relationship.) IaaS is especially suitable as a research topic as the relative homogeneity of the IT resources offered makes it easier to compare decision factors across different enterprises. Moreover, IaaS deserves special attention, as IT infrastructure decisions are usually not

strategic (c.f. [9]). So even if similar determinants guide the adoption decision for different Cloud Computing service models (IaaS, SaaS, Platform-as-a-Service (PaaS)), their relative priority is likely service model-specific.

2 IaaS Adoption Model

2.1 Model Design

Despite the strong media presence of general Cloud Computing, the IaaS concept is far from being clearly defined. Several sources give varying definitions (e.g. Vaquero et al. [29], NIST [25]). According to Hall, information is a determining factor for the diffusion of new technology; the choice of implementing the technology requires knowledge about its existence and its applicability in an enterprise context[17, p.19]. Therefore, proposition 1 is put forward: the unclear definition of IaaS negatively affects IaaS adoption propensity.

The impetus for new technology deployment (e.g. IaaS), mostly builds up in the IT-related areas of any enterprise, as these employees tend to have the technical background knowledge. Many business executives cannot position the current IaaS offerings correctly in respect to the conventional IT sourcing options due to the recency of this technology. Hence, innovation champions from the IT department are key success factors of IaaS adoption (see Rogers [27, p.414]). Proposition 2 maintains: IaaS Innovation champions in the IT departments positively affect IaaS adoption propensity.

As in every sourcing scenario, vendor selection is also an issue for enterprises planning to use IaaS; the challenge here is to assess and select suitable IaaS providers. Among the different provider attributes, absolute size (e.g. in terms of employees, turnover) acts as a signal for trustworthiness. It demonstrates that numerous clients can be served and is hence an expression of provider performance [6, p.466]. Relative size (in terms of market share) suggests the superiority of the services and resources offered by a specific provider[12, p.38]. Provider reputation is also considered a positive attribute[12, p.38]. Proposition 3 therefore assumes: Provider characteristics (relative and absolute size, positive reputation) positively affect IaaS adoption propensity.

Although provider reputation is an important concept for vendor selection, determining the reputation of a certain provider might be difficult and would require the usage of reputation measurement processes and methods. Proposition 4 states: Lacking processes for assessing provider risk and reputation negatively affect IaaS adoption propensity.

The processing of personal data of EU citizens on IT resources located out of EU territory is only permitted if complicated data protection regulations are followed (e.g. the Safe-Harbour treaty between the EU and the USA). Moreover, German data protection laws stipulate that a client has to have control over his data at any time during a commissioned data processing job[23]. This principle collides with the basic premise of IaaS, that the location of the data remains unknown to the client. Parrilli [26] points out that enterprises enjoy almost no legal protection when using Cloud Computing; the business relationship between

an enterprise user and an IaaS provider is solely depending on the contract that those two parties agree on (which might not be fair due to the information asymmetry at play here). Those legal challenges lead to proposition 5: the unfavorable legal situation and binding compliance requirements negatively affect IaaS adoption propensity.

Outsourcing of business data to an IaaS provider generally leads to a certain loss of control over data security and data protection. This problem was first investigated in the context of conventional IT Outsourcing, and was identified as one of the biggest risks of this sourcing option[4, p.92]; it is directly applicable to IaaS. Proposition 6 states: Data security and data protection issues surrounding IaaS negatively affect IaaS adoption propensity.

According to Armbrust et al.[1] and Kim et al.[21], clients' worries about the availability of externally purchased services are the biggest challenges to IaaS providers. Hence, proposition 7 states that concerns about service availability negatively affect IaaS adoption propensity.

Proposition 8 suggests that lacking IaaS monitoring and reporting solutions negatively affect IaaS adoption propensity. This assumption is based on the current situation in IaaS monitoring, which is still in its infancy. There are rudimentary public monitoring services (e.g. CloudStatus[1] or CloudSleuth[2]). However, those offerings are not yet comparable to or integrated into existing in-house monitoring tools.

Missing Application Programming Interfaces (APIs) and incompatible standards are still more the rule than the exception for IaaS offerings. Many providers use proprietary standards for their virtual machine containers and their APIs[21, p.68]. This situation leads to low interoperability among the IaaS providers and hence, current users can become locked-in with one provider. Buyya et al. perceive standardized interfaces of service offerings as indispensable for the success of IaaS Cloud Computing[8, p.7]. Proposition 9 therefore assumes that incompatible standards among IaaS providers negatively affect IaaS adoption propensity.

Price transparency exists if customers can acquire a clear, comprehensive and easily understandable overview of the service tariffs of a provider; especially the comparability of tariffs and benefits across different providers increases customer satisfaction[11, p.309]. When looking at current IaaS offers, customers can hardly compare the individual offerings (e.g. Amazon's ECU (EC2 Compute Unit) vs. Rackspace's concept of guaranteed CPU core percentages). Although tariffs are known, these quality differences lead to price intransparency (c.f. [14]). Moreover, according to Gartner analysts, the Cloud Computing market is on its way from the current pioneer phase to a consolidated market; the complete commoditization is expected from 2015 on[13]. This leads to proposition 10: the difficult cost-benefit evaluation of current IaaS offerings due to price intransparency and market immaturity negatively affects IaaS adoption propensity.

Egle et al. [15] showed in an empirical study that the systematic management of IT costs hitherto only happens for hardware and software expenditures.

[1] http://www.cloudstatus.com/

[2] https://www.cloudsleuth.net/

Communication cost, personnel and operating expenses are often neglected [15, p.12]. As a result, enterprises can state the expenses incurred for certain IT services only very roughly. When sourcing external services, enterprises receive exact information on the costs. Thus, proposition 11 claims that increased internal cost transparency through IaaS usage positively affects IaaS adoption propensity.

Every new technology has to be adapted to the needs of the enterprise, but also, the structure of the enterprise has to be adapted to the new technology [27, p.424]. This adoption process, triggered by innovations, can cause uncertainty and resistance in the organization. In the IaaS context, those actions especially affect employees in IT departments[33, p.134]. Thus, proposition 12 suggests that the unknown organizational impact of IaaS introduction negatively affects IaaS adoption propensity.

The propositions detailed above can be firmly grounded in existing theoretical frameworks. Propositions 3,4,7,8,9 can be derived from agency theory [22], especially from principal-agent-dynamics. Propositions 1,2,10,11,12 are grounded in the concepts of diffusion of innovation theory [27], especially as far as diffusion in organizations is concerned. IT governance theory [30] is applicable to propositions 5 and 6, as they both address accountability and decision rights. Other recognized theories , which are frequently used to explain SaaS adoption (e.g. in [5] or [32]) are not applicable here. The transaction cost theory (c.f. [31]) is not particularly helpful, as the resource specificity of IaaS resources is so low, that only a market-based approach seems reasonable here. The same line of reasoning can be applied to the resource-based view of the enterprise (c.f. [3]). As argued earlier, for most enterprises, IT infrastructure resources can hardly be considered unique capabilities which offer a strategic advantage.

Fig. 1 summarizes the propositions given above and acts as a research model for the following expert interviews.

2.2 Research Method

A qualitative research approach was chosen from the diverse range of available methods. Due to the very early development stage of IaaS and the explorative nature of the research question, qualitative interviews seem to be particularly suitable. In the context of this paper, the guided expert interview was selected; this type of semi-structured interview is regarded as an important basic approach to collecting qualitative data [7, p.308] and allows for open-ended types of questions, as well as for intensive research using small sample counts[7, p.381].

An interview guideline was prepared for the expert interviews; the contents of this guideline reflects the propositions derived in section 2.1. The interview guideline is supposed to serve as an orientation for the interviewer, such that no important detail is forgotten and such that a certain comparability among the collected data is ensured. The research literature considers guideline tests and test interviews as essential[7, p.248]; therefore, three test interviews were conducted and the interview guideline was reviewed through several external specialists. The resulting feedback was incorporated into the guideline.

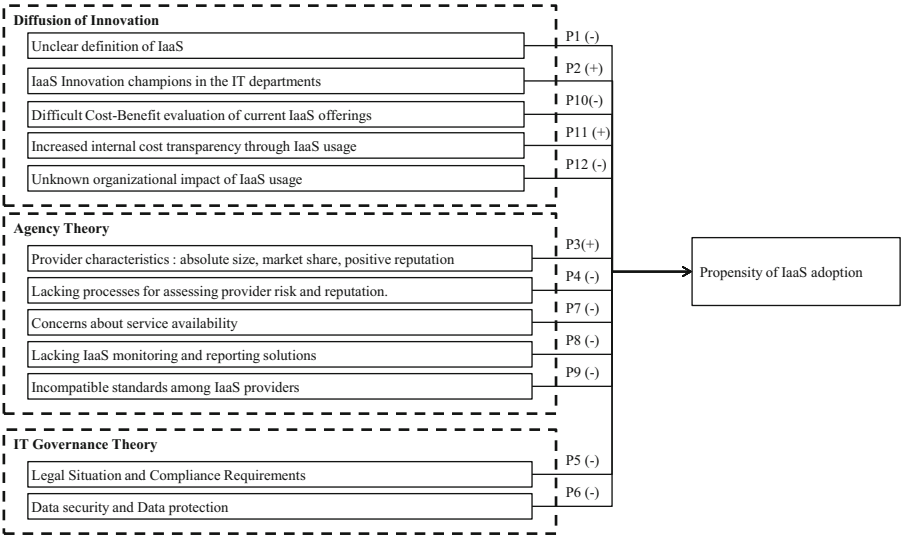


Fig. 1. Research model

Expert interviews are not suited for a large number of survey participants, as the focus lies on the quality and the expressive power of each individual interview [7, p.297]. Research literature recommends sample sizes of 20 to 30 interviews [24, p.441]. It was decided to focus on German enterprises or enterprises with German subsidiaries. German IT executives tend to be conservative towards Cloud Computing (e.g. [19]), so German experts should be especially aware of the potential obstacles to IaaS adoption. The actual experts were searched using the social business network XING[3], which lists the functional skills and the hierarchical position of each participant. Among the chosen experts were IT executives, IT consultants, lawyers specialized in Outsourcing as well as research-oriented enterprises with an IT focus. Further search parameters included experience in the areas of Cloud Computing, Grid Computing, Virtualization, IT Outsourcing and Data Center operations. In total, 215 potential interviewees were identified and invited, out of which ca. 50 experts were willing to participant in an interview. From those, 20 were selected for the actual interviews. They were invited electronically and they were informed that the telephone interviews would take about 20 minutes [7, p.242]; this relatively short time interval was chosen due to the assumed time pressure that these experts are facing. The actual minimum interview length was 15 minutes and the maximum length was 75 minutes; the average interview length was 35 minutes. This rather high variation can be explained by the use of open questions, the target of a free-flowing interview and the specific time pressure of some of the participants. The interviews were conducted during three weeks from April to May 2010.

[3] `http://www.xing.de`

In research literature, the recording of telephone interviews for later interpretation is perceived as indispensable. After the interviews, the written notes and the recorded talks were available for analysis[7, p.311]. Every interview was transcribed word by word using a transcription software for greater clarity in the later interpretation; the transcriptions were then anonymized for data protection reasons and for hiding the expert's identity[7, p.313].

The further analysis follows Meuser and Nagel's method[24]. In a first step, the interviews were paraphrased and then topically ordered. Each passage was associated with a specific headline. In the next step, passages from different interviews, that match topically, were selected and compiled; the associated headlines were harmonized. Then, a conceptualization step showed similarities and differences in the data. The final step, the theoretical generalization, consisted of the inclusion of theories and the arrangement of topics. This step is detailed in the following section 3, as it exhibits the actual results of the interviews.

3 Results

In this section, the summarized results of the expert interviews are given in Tab. 1; the interview transcripts were used to derive hypotheses which will be compared to the propositions from section 2.1 also in Tab. 1; column two shows the original proposition, column three shows the matching hypothesis derived from the expert interviews and the rightmost column shows the percentage of interviewees that supported this hypothesis.

The interview results allowed to derive another hypothesis in addition to the twelve ones described in Tab. 1. This hypothesis can be summarized as follows: "Insufficient Service Level Agreements and Policies in case of lacking service availability negatively affect the introduction and the usage of IaaS". The support of this hypothesis is ca. 20%. This result specifically addresses the practically non-existant penalty payments in case of service failures.

4 Discussion

The results presented in the last section give some interesting insights into IaaS adoption drivers. It becomes clear that the propositions that were developed in section 2.1 were on the whole comprehensive and generally supported by the experts. It is also a sign of external validity, that the general Cloud Computing issues usually noted in surveys like data security and legal issues were also mentioned frequently by the interviewees of this IaaS-centered study. However, the usual potential IaaS benefits like infrastructure agility were not a common topic in the experts' answers, although they were asked for the possible rewards of Cloud Computing.

The most notable discrepancy between the propositions and the interview results becomes apparent in the area of IT-based innovation champions supposedly driving IaaS adoption. The experts interviewed here tend to put the responsibility for the successful IaaS adoption in the hands of management executives, as

Table 1. Summarized Interview Results

Prop. No.	Proposition	Matching Hypothesis	Support of hypothesis
1	Unclear definition of IaaS	proposition supported	60%
2	IaaS Innovation champions in the IT departments	The integration decision of IaaS has to be made according to the requirements of the functional departments in accordance with management.	40%
3	Provider characteristics: size, market share, positive reputation, client references	Prov. characteristics: absolute size, positive reputation, references, further trust-building measures (certifications, data center tours)	55%
4	Lacking processes for assessing provider risk and reputation	proposition supported	15%
5	Legal Situation and Compliance Requirements	proposition supported	40%
6	Data security and Data protection	proposition supported	50%
7	Concerns about service availability	proposition supported	25%
8	Lacking IaaS monitoring and reporting solutions	proposition supported	25%
9	Incompatible standards among IaaS providers	proposition supported	25%
10	Difficult Cost-Benefit evaluation of current IaaS offerings	proposition supported	65%
11	Increased internal cost transparency through IaaS usage	proposition supported	20%
12	Unknown organizational impact of IaaS usage	proposition supported	40%

they are the only ones that can decide to bear the risk of an IaaS implementation. IT departments tend to prepare IaaS adoption decisions according to functional business requirements, but they are not the IaaS innovation champions (perhaps they anticipate the possible organisational consequences).

The top three issues of IaaS adoption are difficult cost-benefit evaluations , the unclear definition of the IaaS concept and the importance of provider characteristics as decisive factors. Every new technology will have to prove its value sooner or later and IaaS makes not exception here (in accordance to Diffusion of Innovation theory). More surprising is the fact, that conceptual difficulties surrounding IaaS prevent its success; this result has not been visible in other recent surveys on Cloud Computing (e.g. those mentioned in Sec. 1.1). It seems that IaaS adoption depends on the dissemination of a clearer notion of the concept in organizations, especially to a non-technical audience. The third major issue is the

provider-customer relationship, especially the creation of trust among the parties involved. Conventional and well understood trust-inducing signals as market size and reputation are important, but further trust-building measures (like certifications e.g. ISO27001 or data center tours) are essential for IaaS adoption.

This research approach generated a number of interesting hypotheses, that warrant further investigation, especially the role of trust in the provider-client relationship and the role of the IT department in IaaS adoption. Those hypotheses would be an ideal target for a quantitative empirical research study.

4.1 Limitations on the Research Design and Material

The research method utilized here shows some inherent weaknesses. Telephone interviews are perceived as unfavorable in the research literature, because of their impersonal character and possible, not controllable circumstances (e.g. distractions) during the call[7, p.242]. Moreover, expert selection using a social network entails the risk of making poor choices, as qualifications of the social network participants cannot be verified.

An inherent weakness arises through the non-standardized IaaS nomenclature; therefore, a certain conceptual fuzziness is introduced into the investigation, as every expert probably had a slightly different notion of the IaaS concept.

4.2 Conclusion

This paper aimed at investigating the organizational factors of IaaS adoption, as this issues is of high practical relevance and as the related scientific literature failed to answer this relevant question in a sufficient way so far. Towards this end, an IaaS adoption model containing adoption drivers and deterrents was proposed, based on an appropriate theoretical foundation. The model served as an input to rigorously planned expert interviews following scientific best-practices. The propositions of the adoption model were generally supported by the experts, however, the role of trust and executive involvement were underestimated in the initial propositions. As a result, missing trust, intransparency in IaaS offerings and conceptual unclarity of IaaS can be assumed to be decisive issues for IaaS adoption.

References

1. Armbrust, M., Fox, A., Joseph, A.D., Katz, R.H., Konwinski, A., Lee, G., Patterson, D.A., Rabkin, A., Stoica, I., Zaharia, M.: Above the Clouds: A Berkeley View of Cloud Computing. Technical Report UCB/EECS-2009-28, University of California at Berkeley (February 2009), `http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.pdf`
2. Baier, M.: Die Diffusion von Innovationen im Markt managen: Fallstudie zur Nutzung von Grid-Technologien für betriebliche Informationssysteme (BIS). In: Bichler, M., Hess, T., Krcmar, H., Lechner, U., Matthes, F., Picot, A., Speitkamp, B., Wolf, P. (eds.) Multikonferenz Wirtschaftsinformatik 2008, GITO-Verlag, pp. 104–114. Berlin (2008)

3. Barney, J.: Firm resources and sustained competitive advantage. Journal of Management 17(1), 99–120 (1991)
4. Barthelemy, J., Adsit, D.: The seven deadly sins of outsourcing. The Academy of Management Executive 17(2), 87–100 (1993)
5. Benlian, A., Hess, T., Buxmann, P.: Drivers of SaaS-Adoption – An Empirical Study of Different Application Types. Bus. Inf. Syst. Eng. 5, 414–428 (2009)
6. Bensaou, M., Anderson, E.: Buyer-supplier relations in industrial markets: When do buyers risk making idiosyncratic investments? Organization Science 10(4), 460–481 (1999)
7. Bortz, J., Döring, N.: Forschungsmethoden und Evaluation: für Human- und Sozialwissenschaftler [Research methods and Evaluation for Social scientists], 4th edn. Springer, Berlin (2006)
8. Buyya, R., Yeo, C.S., Venugopal, S.: Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities. In: Proceedings of the 10th IEEE International Conference on High Performance Computing and Communications, Dalian, China (2008)
9. Carr, N.G.: It doesn't matter. Harvard Business Review (May 2003)
10. CIO Magazine: Cloud computing survey (June 2009), http://www.cio.com/documents/whitepapers/CIOCloudComputingSurveyJune2009V3.pdf (last accessed May 13, 2010)
11. Diller, H., Herrmann, A.: Handbuch Preispolitik [Handbook on Pricing], 1st edn. Gabler, Wiesbaden (2003)
12. Doney, P.M., Cannon, J.P.: An examination of the nature of trust in buyer-seller relationships. The Journal of Marketing 61(2), 35–51 (1997)
13. Driver, M.: Cloud application infrastructure technologies need seven years to mature. Research report G00162990, Gartner Inc., Stamford, USA (2008)
14. Durkee, D.: Why cloud computing will never be free. Queue 8(4), 20–29 (2010)
15. Egle, U., Weibel, D., Myrach, T.: Ziele und erfasste Kosten im IT-Kostenmanagement: Eine empirische Untersuchung.[Targets and recorded costs in the IT cost management]. In: Bichler, M., Hess, T., Krcmar, H., Lechner, U., Matthes, F., Picot, A., Speitkamp, B., Wolf, P. (eds.) Multikonferenz Wirtschaftsinformatik 2008, GITO-Verlag (2008)
16. Gartner Inc.: Hype cycle for cloud computing, 2009. Research Report G00168780, Gartner Inc., Stamford, USA (July 2009)
17. Hall, B.H.: Innovation and Diffusion. In: The Oxford Handbook of Innovation, ch. 17, pp. 459–484. Oxford University Press, Oxford (2005)
18. Hwang, J., Park, J.: Decision factors of enterprises for adopting grid computing. In: Veit, D.J., Altmann, J. (eds.) GECON 2007. LNCS, vol. 4685, pp. 16–28. Springer, Heidelberg (2007)
19. IDC Central Europe: IDC-Studie: Cloud Computing in Deutschland ist noch nicht angekommen (June 2009), http://www.idc.com/germany/press/presse_cloudcomp.jsp, (last accessed October 7, 2009)
20. Jayatilaka, B., Schwarz, A., Hirschheim, R.: Determinants of ASP choice: an integrated perspective. European Journal of Information Systems 12(3), 210–224 (2003)
21. Kim, W., Kim, S.D., Lee, E., Lee, S.: Adoption issues for cloud computing. In: The 11th International Conference on Information Integration and Web-based Applications & Services(iiWAS 2009), Kuala Lumpur (December 2009), http://uclab.khu.ac.kr/resources/publication/C_196.pdf (last accessed April 08, 2010)

22. Logan, M.S.: Using agency theory to design successful outsourcing relationships. The International Journal of Logistics Management 11(2), 21–32 (2000)
23. Meents, J.G.: Cloud computing: Rechtlich aspekte [legal aspects] (May 2010), http://www.haufe.de/recht/newsDetails?newsID=1272627923.17 (last accessed May 25 2010)
24. Meuser, M., Nagel, U.: Das Experteninterview: Konzeptionelle Grundlagen und methodische Anlage [The expert interview]. In: Methoden der vergleichenden Politik- und Sozialwissenschaft, pp. 465–479. VS Verlag, Wiesbaden (2009)
25. NIST: Draft NIST Working Definition of Cloud Computing (August 2009), http://csrc.nist.gov/groups/SNS/cloud-computing/cloud-def-v15.doc (last accessed November 20, 2009)
26. Parrilli, D.M.: The determination of jurisdiction in grid and cloud service level agreements. In: Altmann, J., Buyya, R., Rana, O.F. (eds.) GECON 2009. LNCS, vol. 5745, pp. 128–139. Springer, Heidelberg (2009)
27. Rogers, E.M.: Diffusion of Innovations, 5th edn. Free Press, New York (August 2003)
28. Thanos, G.A., Courcoubetis, C., Stamoulis, G.D.: Adopting the grid for business purposes: The main objectives and the associated economic issues. In: Veit, D.J., Altmann, J. (eds.) GECON 2007. LNCS, vol. 4685, pp. 1–15. Springer, Heidelberg (2007)
29. Vaquero, L.M., Rodero-Merino, L., Caceres, J., Lindner, M.: A break in the clouds: towards a cloud definition. SIGCOMM Comput. Commun. Rev. 39(1), 50–55 (2009)
30. Weill, P., Ross, J.: IT Governance. Harvard Busines School Press, Boston (2004)
31. Williamson, O.E.: The economic institutions of capitalism: firms, markets, relational contracting. The Free Press, New York (1985)
32. Xin, M., Levina, N.: Software-as-a service model: Elaborating client-side adoption factors. In: ICIS 2008 Proceedings (2008), http://aisel.aisnet.org/icis2008/86 (paper 86)
33. Yanosky, R.: The Tower and the Cloud, chap. From Users to Choosers: The Cloud and the Changing Shape of Enterprise Authority, pp. 126 – 136. Educause (2008)

ETSI CLOUD – Initial Standardization Requirements for Cloud Services

Karsten Oberle[1] and Mike Fisher[2]

[1] Alcatel-Lucent Bell Labs, ETSI TC CLOUD Vice Chairman
Lorenzstrasse 10, 70435 Stuttgart, Germany
karsten.oberle@alcatel-lucent.com
[2] BT Innovate and Design, ETSI TC CLOUD Chairman
BT Adastral Park, Ipswich IP5 3RE, UK
mike.fisher@bt.com

Abstract. While the technological basis for cloud services is relatively mature, the development of the market is still at an early stage. There is considerable potential, but also a number of widely held concerns which are inhibiting mainstream adoption of cloud services by business. This paper is based on the outcome of the ETSI TC CLOUD Workshop, "Grids, Clouds and Service Infrastructures", an event which brought together key stakeholders of the grid, cloud and telecommunication domains to review the state of the art and current trends. The focus was on areas where standardization has the potential to encourage development of the market, with a particular focus on cloud computing and services. This paper introduces and expands on the conclusions reached. It is intended to serve as the basis for future work.

Keywords: Standardization, ETSI, Cloud services.

1 Introduction

In recent years the emergence of the Internet as a communications platform offering near universal connectivity has led to significant changes in the way that IT systems are designed and operated. Continuing a trend that started with virtual private networks for enterprises, the emphasis is clearly shifting away from dedicated computing resources and networks to support specific applications. More flexible approaches which promise greater resilience to changing economic and technical circumstances are attracting a high level of interest. Grid computing has built on earlier advances in distributed systems to allow effective sharing of computing resources, particularly in the scientific applications for which grids were designed. The combination of grid technologies with Service Oriented Architecture forms the basis of Service Oriented Infrastructure which is gaining acceptance in mainstream enterprise IT systems. This offers a solution to long-standing business needs for efficient use of computing resources, with reduced costs. It also enables construction of flexible ICT infrastructure which supports the growing need to be able to make rapid changes to systems and processes to respond in a competitive business environment.

J. Altmann and O.F. Rana (Eds.): GECON 2010, LNCS 6296, pp. 105–115, 2010.

Combining Internet connectivity with highly scalable hosted computing services, generally with usage-based charging, is now being referred to as cloud computing. This has been supported by the increased flexibility offered by virtualisation of computing resources and the significant economies of scale achievable in very large data centres [1]. Major investment by providers of computing and storage resources has stimulated considerable interest in this topic which is seen as the next step in the development of a future service-oriented ICT infrastructure.

Telecommunications operators are beginning to expose more advanced capabilities of their networks for use in both internal and third party services, generally following service-oriented principles. As the telecom and IT worlds converge, radical changes in the way that ICT infrastructure is provided and used on a global scale are anticipated. The promise of cost-effective and flexible access to highly scalable resources offered by infrastructure providers in a competitive market could allow enterprises, including SMEs, to offer services based on cloud resources with low initial cost but with the ability to rapidly scale up in response to surges in demand.

Although early entrants to the market are already providing significant value to their customers and attracting considerable interest, overcoming barriers such as software portability and interoperability between infrastructure providers is required if the full potential of the cloud vision is to be realised. Users of cloud resources would then be able to choose among different service providers according to their needs, including performance, dependability, geographical spread and cost. This will require appropriate standards to be developed, with broad consensus building across industry sectors, public authorities and academia. Some work in this direction is already beginning in bodies such as ETSI TC CLOUD [2], OGF-Europe [3] and DMTF [4].

A Workshop on "Grids, Clouds and Service Infrastructures" [5] organized by ETSI TC CLOUD brought together experts from industry (Telecom and IT), research and standardization to explore emerging trends, to promote common approaches and to prepare the way for cooperative standards development. This paper summarizes the outcome of the workshop and presents some adoption issues which could be addressed by the development of open standards.

2 Cloud Computing – An Introduction

The term "cloud computing" is applied to a range of different approaches to delivering IT capabilities over networks, typically the Internet. It originates from the use of a cloud to represent wide area networks in diagrams – indicating that the details of how data are transported are hidden from the endpoints. Only correct delivery is significant to the end user. The network provider has the freedom to configure its systems and operations to meet its own business goals. Cloud computing represents the extension of this general idea to include a wider range of networked IT components such as servers, storage and data resources.

The National Institute of Standards and Technology (NIST) has recently proposed a definition of cloud computing which is becoming generally accepted:

"Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction." [6]

Various services based on cloud computing infrastructures have emerged recently. From the customers‟ point of view, much of their attraction lies in the ability to purchase only what they need (e.g. infrastructure, value-added platforms or software packages), without having to plan far ahead. There is typically little up-front commitment and the potential ability to achieve flexible scaling to meet dynamic demand. This can enable "pay-as-you-grow" commercial models.

The shift to utility models such as cloud computing can be compared with the move in telecommunications from private circuits to virtual private networks (VPN). Moving from private, dedicated infrastructure to a managed shared service promises clear cost benefits to the customer, provided that issues such as performance, availability and security are predictable and meet their needs. From a provider‟s point of view, sharing of resources between customers and bulk purchasing can lead to economies of scale. Commercial trends in computing infrastructure have reached a point where these economies of scale can make the provision of computing infrastructure as a utility viable. However, mainstream adoption of cloud services is currently limited, at least in part due to the technological diversity of current offerings in terms of, for example, different virtualization technologies or different interface definitions to the services. The following sections describe a number of issues affecting adoption of cloud services where development of standards could play a role. This paper does not set out a specific agenda for standardization. Some issues may be better addressed in other ways, others may require consensus that is not feasible to achieve at present. However, the aim is for this paper to set the context for more detailed discussion aimed at stimulating the market for cloud services.

3 Standards Requirements for Cloud Computing

3.1 Portability

Portability in general refers to the ability to migrate applications between different clouds. This is required to allow customers of cloud services to avoid the situation of being locked into a specific cloud infrastructure provider, having made the decision to run an application in the cloud. This adds to the perceived risks associated with moving to cloud computing. A potential customer needs a high level of trust in the technical and commercial ability of a chosen provider to support critical business applications in the long term.

Current cloud infrastructure providers offer their own proprietary interfaces to application developers. Standardised interfaces to manage cloud infrastructures and the different types of resource they provide are required. Reducing the mismatch between different cloud infrastructure systems would not only enable a competitive market but also enable new business models where different cloud infrastructures can be traded according to price and demand.

At one extreme, the ability to automatically migrate a complete running application (including any necessary monitoring and management features) from one cloud infrastructure to another would clearly be attractive to customers (but much less so to cloud infrastructure providers, particularly in the current market).

Portability of a virtual machine images is being addressed by the DMTF Open Virtualisation Format (OVF) [7]. This should provide a good basis for limited portability but does not address complex configuration or interactions with any supporting systems. These issues are naturally within the scope of a service provider.

Portability of data is essential for a typical business application. The ability for a customer to retrieve application data from one cloud infrastructure provider and import this into an equivalent application hosted by an alternative provider reduces the risk of long-term dependency. This would probably involve a not insignificant amount of effort from the customer (or an application service provider - the details will be largely application specific and not the responsibility of a cloud infrastructure provider). Achieving data portability depends on effective standardization of data import and export functionality between cloud infrastructure providers.

3.2 Interoperability of Clouds

Interoperability is closely related to portability. Here we interpret interoperability as the ability to federate multiple clouds to support a single application. In other words, interoperability involves software and data simultaneously active in more than one cloud infrastructure, interacting to serve a common purpose. This may be motivated, for example, by the availability of specialized resources (including data) from certain providers, or to provide applications in different geographical areas where performance or regulatory constraints are best satisfied by using multiple clouds.

Considering cloud interoperability in general, the term "Intercloud", possibly coined by Cisco, is starting to gain some acceptance. It is analogous to the Internet and based on a similar vision – connecting individual, essentially uncoordinated cloud infrastructures and giving control to the users. One of the lessons of the Internet is that the use of common protocols (or interfaces) readily accessible to developers has great benefits in stimulating innovation. In a recent lecture, Vinton Cerf discussed the current state of play [8]:

"Each cloud is a system unto itself. There is no way to express the idea of exchanging information between distinct computing clouds because there is no way to express the idea of „another cloud". Nor is there any way to describe the information that is to be exchanged. Moreover, if the information contained in one computing cloud is protected from access by any but authorized users, there is no way to express how that protection is provided and how information about it should be propagated to another cloud when the data is transferred"

There are two obvious scenarios that have to be considered when an application is supported by a federation across multiple clouds. In the first scenario the application is managed by an entity, which may be the end user, which interacts individually with each cloud provider. This entity is directly responsible for coordination between the cloud service providers and manages the application via standardised management and monitoring interfaces. In the second scenario a cloud service provider may take on this coordination role, using services from a number of third party providers to meet the overall performance guarantee to the end user. One example could be a compute cloud provider using a network provider for delivery. This scenario requires standards for the sharing of management information such as SLA goals between service providers without exposing too much confidential detail of how the goals are met.

Even these two quite obvious scenarios allow us to suggest a number of dimensions for the discussion of Cloud interoperability:

- Application/Service: Interoperability standards should support distributed applications with predictable behaviour and performance. Components of a single application could be deployed across multiple cloud infrastructure providers and possibly reconfigured while running, or with limited interruption, to respond to changes in usage patterns or resource availability, for example. Application configuration must be resilient to changes in the configuration within each cloud – for example scaling or migration of computational resources.
- Management: Standardised interfaces should be provided by cloud service providers so that a single application can be managed in a consistent way, end-to-end – substantially independent of the details of its deployment across multiple cloud infrastructures. It must be possible for a management application to control and coordinate components in multiple clouds. Standardised management functionality for deployment and migration of virtual machine images between different cloud infrastructures is required. Management interoperability requires interactions between multiple independent actors responsible for application management and infrastructure management.
- Data: Standards are required to support the deployment of equivalent virtual machine images and application data to different cloud infrastructures. A basic requirement for cloud interoperability is network connectivity between cloud environments, appropriate to carry application traffic. In particular, security and other cross-domain data management issues need to be handled in standard ways.
- Network aspects: Network access to computational resources is fundamental to cloud services. Standards are required to support both uniform access to individual cloud computing resources and concatenation or federation of clouds in different locations. Interconnection between clouds should support the quality requirements of applications. Network connectivity will in general be based on the use of shared physical resources in a similar way to computing and storage. Standards for allocation and admission control will be needed.

3.3 Closer Integration of IT and Network Resources

The anticipated convergence of computing, storage and networking into a single integrated infrastructure is becoming a reality. The distinction between network services and the applications they support is disappearing. From the user‟s point of view, the application experience is what matters and this depends on all the supporting systems performing effectively. This closer integration of IT and network resources is of special interest for real-time and interactive applications (e.g. from the telecommunications area) with particular requirements on network performance. Deploying these applications in the cloud will become feasible when IT and network resources are integrated as a unified infrastructure.

The trend towards cloud computing is breaking the association between application software and physical hardware in the interests of flexibility and resource

efficiency – a shift already underway as a result of server virtualisation. As hardware configurations in complex applications change dynamically, the task of keeping all the endpoints connected becomes increasingly complicated.

A consistent approach to automation of ICT infrastructure is required. Each component (computing, storage and network) needs to be dynamically configurable to respond to changes in the performance or configuration of other elements. Effective monitoring and control solutions are needed, which allow the automation and associated cost benefits offered by cloud computing infrastructures to be extended across the whole infrastructure involved in supporting applications.

3.4 APIs to Networking/Data Movement Functionality

This requirement is closely related to topics covered in sections 3.2, 3.3 and 3.7. Today"s value-added/platform cloud service providers (i.e. those offering support for particular types of application rather than basic resource infrastructure) offer relatively static resource management, typically from a single infrastructure provider. Resources such as computing, storage and network are available to their customers (e.g. application providers) from several different providers and they will increasingly expect to be able to make use of the full range of available infrastructure in a dynamic way. This requires the ability for a distributed application to configure or adapt to the resources available to it at runtime. Current efforts to allow applications to interact with the (virtual) computing resources available to them include, for example, scaling by changing the amount of computing resource allocated to an image, or by adding additional virtual machines. Such mechanisms need to be supplemented with functionality to allow efficient and adaptive movement of data over the networks supporting a distributed application. This could be facilitated by providing suitable APIs. They should offer the ability to discover available network connectivity and storage resources, and for applications to use them effectively.

3.5 Support for Building, Modelling, Testing and Deploying Applications

To promote the wider acceptance of cloud platforms by application providers, it is necessary to make it easier to use those platforms to achieve dependable applications. There should be tool support for application providers giving them the ability to efficiently develop, deploy and manage their applications. It is not clear to what extent such tools will require standardization (e.g. APIs), but the ability to target multiple platforms from a common toolset may be attractive. Examples of service engineering and support functionality which would be useful in cloud platforms include:

- Performance estimation for components of loosely coupled applications
- Benchmarks to characterise and verify performance parameters of application components
- Monitoring of application performance and relation to infrastructure level quality parameters
- Mapping of high level application descriptions to resource requirements

The aim is to help an application developer and provider to understand how to construct applications in a modular way and deploy onto a cloud infrastructure, to

characterise individual components, to model the expected behaviour and validate the results of modelling in controlled test environments prior to live deployment. Once an application is deployed, appropriate monitoring is required to verify that the application behaves as anticipated.

3.6 Support for Optimisation of Distributed Applications

The typical expectations of a customer of cloud services are that their applications deliver well-defined quality levels to end users, independent of the load on the application (e.g. number of active users) and at a price proportional to the load. Application software and data is deployed into one or more cloud infrastructures according to anticipated usage patterns, taking into account expected availability of resources (computing, storage and network) which are shared with other applications. It may be necessary to adjust the deployment of an application if usage patterns or resource availability change.

It is desirable that the stakeholders (e.g. cloud service providers, application provider) responsible for this deployment can make informed decisions to achieve good resource utilisation and assure application quality. This requires knowledge of characteristics of both the application and the cloud infrastructure. Standardization of the description of these characteristics could play an important role, particularly where an application is not restricted to a single cloud (e.g. hybrid private/public deployments, cloud portability or interoperability scenarios).

3.7 Clearly Defined SLAs, Fit for Business Use

Service Level Agreements (SLAs) in this context are understood to be unambiguous statements of the expectations and responsibilities of both users and providers of cloud services. They are an important part of established approaches to Service Level Management used in ITIL [9] and the TM Forum [10].

An SLA is a contract between the provider and the customer of a service specifying the function performed by the service, the agreed bounds of performance, the obligations on both parties to the contract and how any deviations are to be handled. An SLA is made in a business context and therefore it must include all aspects of the interaction between the provider and customer relevant to the service.

There are two distinct requirements for standardization relating to SLAs for cloud services.

- SLAs (or SLA templates, representing an invitation to treat) need representations which are clearly understood by both parties, make explicit the expectations and obligations on each party and can be used to compare different providers.
- A means of mapping between higher level and lower level requirements in a clearly defined way would also be desirable. This could support some form of end-to-end negotiation in cases where a service involves multiple stakeholders and independent bipartite interactions prove insufficient to assure end-to-end service behaviour.

In order to match the technical characteristics of cloud infrastructures with an appropriate commercial environment, a flexible, lightweight and dynamic framework for management throughout the SLA lifecycle is required.

3.8 Data Protection, Privacy, and Security in Clouds

This refers to the data protection, privacy and security issues associated with the use of cloud services for sensitive applications. It encompasses processing and storing of personal or otherwise sensitive (such as business sensitive data) data. Processing, using or storing data in clouds is widely perceived as introducing new risks to customers of cloud services. The abstraction associated with the use of cloud infrastructure can result in reduced control by the owner of the data. Widespread adoption of cloud services for enterprise applications will require confidence that potentially sensitive business data is handled appropriately – governed by policies set by the owner of the data. In addition, distribution of responsibilities concerning the processing and storing of personal data is an area that needs to be addressed both on a technical and regulatory level.

Privacy concerns the right of natural persons to not be subject to identity crime, tracking, or other undesired and unlawful intervention concerning an individual‟s identity or behaviour.

Data protection addresses the aspect of preventing undesired disclosure or manipulation of personal or otherwise sensitive information.

Security covers confidentiality, integrity, availability, authenticity, accountability and non-repudiation.

Cloud infrastructure is based on the principle of sharing and geographical distribution of resources. This enables a new attack venue, as data from different legal entities share the same distributed infrastructure. Sensitive data from several entities may be physically co-located, although logically separated. Sensitive data from a single entity may be spread across several physical locations with different security policies. These are only a few of the many privacy, data protection and security aspects of importance that may be critical for the wider adoption of cloud computing. There are also regulatory aspects putting constraints on cloud adoption, such as the EU directives 2002/58/EC [11] “The processing of personal data and the protection of privacy in the electronic communications sector (Directive on privacy and electronic communications)” and 95/46/EC [12] “The protection of individuals with regard to the processing of personal data and on the free movement of such data”.

Topics of interest include:

- Protection against communications threats and risks
- Identification, authentication and authorization of users, providers and services
- Service and data protection
- Protection against malicious communication
- Secure storage
- Lawful Interception (LI)
- Data retention

3.9 Regulatory Aspects

Regulation has a broader impact on cloud services than just Data Protection and Privacy (DPP). Some current approaches (e.g. EU Directives) include constraints based on the location and control of data. It is not clear how location is defined in cloud services. Answering this question may require exposing some of the details of how cloud infrastructure is built – in terms of data centre locations and the way that storage and transport of data is managed (placed, distributed, replicated, cached,…), for example. Cloud service providers are likely to be reluctant to share this information openly but mechanisms for demonstrating compliance with relevant legislation, acceptable to regulatory authorities, are required. In addition, accountability for security and privacy of personal data needs to be properly defined – in a cloud deployment, what is the balance of responsibility between the owner of the data and the cloud service provider? The international nature of cloud infrastructures means that these issues require solutions which work across jurisdictions.

3.10 Support for Demanding Applications (e.g., Interactive Media)

One of the main benefits of cloud technologies is that of transparency; the end user neither knows nor cares where in the cloud his application executes as long as the SLAs are maintained. There are a number of existing and proposed consumer application areas such as media transformation where the use of cloud services can have a significant impact but the performance requirements are particularly demanding. New approaches to management may be required to address these requirements.

The Internet is increasingly used as a media delivery system for video and audio streams (both live and pre –recorded). There is an increasing interest in the consumer market in interactive media such as online gaming with dynamically rendered environments.

At the same time, the variety of terminal devices to consume this content is growing rapidly – from large screen (with 3D now emerging) displays connected to powerful computers and broadband networks to mobile phones with much smaller screens and much more constrained computing power and network connectivity. Transcoding of media to make it suitable for delivery to and display by such varied devices is challenging, particularly if it needs to be done in close to real time.

Such applications involve several cloud resources in their end-to-end operation, with implications for computing, storage and network performance. Consumers of networked media are geographically distributed and require encoding and delivery that is appropriate to their terminal equipment. Transcoding from an original base format to meet the needs of diverse users is computationally intensive. So too is dynamic rendering of shared virtual environments. Caching techniques or logistical networking may allow the results to be reused by many consumers, at the cost of additional storage. Multicast networking may also have a role. High definition or 3D TV or video generally involves very large volumes of data, presenting challenges for storage (both primary and staging) and network capacity. The inertia of data imposes severe restrictions on the feasibility of transcoding as a cloud service and further highlights the need to consider computing, storage and network resources together.

Optimised media creation, adaptation, archiving, distribution and delivery based on the use of cloud infrastructure could be an attractive Platform-as-a-Service (PaaS) offering with scope for standardising interfaces to common features.

3.11 Software Licensing

Software licensing is a major inhibitor of the adoption of flexible computing models, including cloud infrastructure services. Cost savings in hardware, IT infrastructure management and energy can be negated by the need to purchase sufficient licences to cover the maximum size of an application deployment in advance. The basis for software licensing varies considerably between vendors. Costs may be associated with the (maximum) number of servers, CPUs or cores the software is deployed on. Licence management systems are available to track and enforce this kind of usage. They may be related to the total or maximum concurrent number of users. Less common arrangements include restrictions to a geographical area (e.g. on a single site) or to a limited set of named users. Also limiting flexibility are restrictions to running on specific physical hardware – requiring reactivation when transferred. Most of these licensing schemes are based on assumptions about the kind of physical infrastructure used to run the application in question. It is therefore not surprising that issues arise when new approaches to computing infrastructure, such as cloud, begin to emerge.

Software vendors are unlikely to change licensing models unless they see new market opportunities or face a competitive threat. Standardization does not have an obvious role in their business decisions. However, technical solutions to licence management could increase the flexibility and the range of viable options for a software vendor. For example, the ability for software licensed to a particular user to run in an external cloud environment on behalf of that user would be desirable. This might include the ability to track concurrent use in both private and public cloud environments. Details of the use of licensed software and the mechanisms to audit and enforce licence terms in a cloud environment need to be specified in cloud service SLAs.

4 Conclusions

Cloud services promise to radically change the way that software-intensive systems are deployed and managed. While many of the necessary technological building blocks already exist, the market and economic basis for an ecosystem of service providers and consumers are still at an early stage of development. Some early successes have stimulated considerable interest in cloud services to support mainstream enterprise systems. However, there remain several areas of concern which are inhibiting their adoption. This paper lists and describes some of these where development of standards might contribute to a solution. The intention is that this will form the basis for future work to analyse and prioritise requirements for standards to support the emerging market for cloud services.

Acknowledgement

This paper is based on an ETSI Technical Report [13]. The authors acknowledge the members of ETSI TC CLOUD and TISPAN who contributed to and reviewed its content. The authors also gratefully thank all those involved in the "Grids, Clouds and Service Infrastructures" workshop [5] for their active and stimulating participation.

References

1. Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R.H., Konwinski, A., Lee, G., Patterson, D.A., Rabkin, A., Stoica, I., Zaharia, M.: Above the Clouds: A Berkeley View of Cloud Computing, http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.pdf
2. European Telecommunications Standards Institute (ETSI): TC CLOUD, http://portal.etsi.org/portal/server.pt/community/CLOUD/310
3. Open Grid Forum (OGF) Europe, http://www.ogfeurope.eu/
4. Distributed Management Task Force (DMTF), http://www.dmtf.org/home
5. ETSI: Workshop on Grids, Clouds and Service Infrastructures, Sophia Antipolis, France (2009), http://www.etsi.org/WebSite/NewsandEvents/Past_Events/2009_GRIDWORKSHOP.aspx
6. Mell, P., Grance, T.: The NIST Definition of Cloud Computing, Version 15 (2009), http://csrc.nist.gov/groups/SNS/cloud-computing/
7. Open Virtualization Format Specification, DMTF Document Number DSP0243 (2010)
8. Cerf, V.: Cloud Computing and the Internet, Google Research Blog (2009), http://googleresearch.blogspot.com/2009/04/cloud-computing-and-internet.html
9. ITIL, Office of Government Commerce, http://www.ogc.gov.uk/guidance_itil.asp
10. SLA Handbook Solution Suite Version 2.0, TM Forum, GB 917 Version 2, http://www.tmforum.org
11. Directive 2002/58/EC of the European Parliament and of the Council of 12 July 2002 concerning the processing of personal data and the protection of privacy in the electronic communications sector (Directive on privacy and electronic communications) (2002)
12. Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing or personal data and on the free movement of such data (1995)
13. European Telecommunications Standards Institute (ETSI): Technical Report, ETSI TR 102 997, v1.1.1 (2010)

Approaching the Internalization Challenge of Grid Technologies into e-Society by e-Human "Grid" Ecology

Tobias A. Knoch[1,3,*], Volkmar Baumgärtner[4], Frank G. Grosveld[2], and Kurt Egger[5]

[1] Biophysical Genomics & Erasmus Computing Grid,
[2] Dept. Cell Biology & Genetics, Erasmus MC, Dr. Molewaterplein 50, 3015 GE Rotterdam, The Netherlands
[3] Biophysical Genomics, Genome Organization & Function, BioQuant Centre/ German Cancer Research Center, Im Neuenheimer Feld 267, 69120 Heidelberg, Germany
[4] Regionalverband Mittlerer Oberrhein, Baumeisterstr. 2, 76137 Karlsruhe, Germany
[5] Emeritus Institute for Plant Sciences, University of Heidelberg Pleikartsförsterhof 2, 69124 Heidelberg, Germany
TA.Knoch@taknoch.org, Volkmar.Baumgaertner@region-karlsruhe.de, F.Grosveld@erasmusmc.nl, KEgger@hip.uni-heidelberg.de

Abstract. The amount of information is growing exponentially with ever-new technologies emerging and is believed to be always at the limit. In contrast, huge resources are obviously available, which are underused in the IT sector, similar as e.g. in the renewable energy sector. This is especially for grid with its fast turnover rates very astonishing considering the barriers for further development put forward by the inability to satisfy the need for such resources. The phenomenon is a typical example of the *Inverse Tragedy of the Commons*, i.e. resources are underexploited in contrast to the unsustainable and destructing overexploitation in the *Classic Tragedy of the Commons*. An analysis of IT and the grid sector which attempts to share resources for better usage efficiency, reveals two challenges, which lead to the heart of the paradox: i) From a macro perspective all grid infrastructures involve not only mere technical solutions but also dominantly all of the autopoietic social sub-systems ranging from religion to policy. ii) On the micro level the individual players and their psychology and risk behaviour are of major importance for acting within the macro autopoietic framework. Consequently, the challenges of grid implementation are similar to those of other pressing global issues as e.g. climate protection. This is well described by extending the *Human Ecology* triangle to a rectangle: invironment-individual-society-environment. By applying this extension of this classical field of interdisciplinary basic and applied research to the grid sector, i.e. by further extension to an *e-Human Grid Ecology* rational, the *Grid Inverse Tragedy of the Commons* can be understood and approached regarding the internalization challenge into e-Society and e-Life, from which then guidelines for the day-to-day management can be derived. This is of general importance for many complex fields and thus with similar paradoxes and challenges.

* Corresponding author.

J. Altmann and O.F. Rana (Eds.): GECON 2010, LNCS 6296, pp. 116–128, 2010.

Keywords: Grid infrastructures, inverse tragedy of the commons, autopoietic sub-systems, risk deep psychology, e-society, e-human grid ecology.

1 Introduction

With ever-new technologies emerging from R&D also the information amount to be stored and processed is exponentially growing and thus believed to be always at the limit. IT has become the key to success in modern life: R&D is meanwhile mostly based on the storage and analysis of huge data amounts, whether for elementary particle physics or the industrial *in silicio* design of cars, aircrafts or entire production facilities. In health care, diagnosis and treatment rely on imaging facilities, their sophisticated analysis and treatment planning. In logistics, the shipment of goods, water, electricity and fuels is driven by efficient distribution management systems. Beyond, the financial and risk management sectors are unthinkable without modeling. Finally, the IT sector itself is inevitably carried by the creation and manipulation of data streams. Thus, currently, the demands outweigh the useable resources by far and especially the public sector struggles to increase the accessible IT resources.

Limits showing e.g. syntropic/entropic materialistic, energetic or other barriers as those of the IT sector, are well known [1, 2]. Such limits have constrained life since its beginning and are one of the evolutionary drivers by the "survival of the fittest". Exponential demand growth until reaching a limit seems to be an inherent property of life and evolution in general [3]. The other side of demand growth – waste and pollution – complies with this similarly, although it is not using a resource but destroying the purity of another one. Obviously, this sustainability challenge beyond the materialistic regime can be found on all evolutionary levels up to the psychological, societal and cultural level. All these levels act as a possible cause for exponential growth.

Especially, the abilities of man in his modern societies have accelerated the use of common resources tremendously reaching the planetary carrying capacity [4]. Climate change and the sustainability challenge, thus is a complex combination of various effects, which in their holistic consequences have reached an unsustainable level threatening survival. The (*Classic*) *Tragedy of the Commons* [5-10] describes this dilemma, in which (multiple) independently acting individuals due to their own self-interest can ultimately destroy a shared limited resource even though it is clear that it is not in the long-term interest of the community locally or for the society in general.

On universal time scales syntropy/entropy laws obviously predict that mankind will reach fundamental limits [3]. Nevertheless, on short time scales huge resources are available for development: Due to the pervasiveness of PCs, their number has grown >1 billion world-wide, outweighing that of specialized computing centers by a factor >100. Since the capacity is peak performance oriented, less than 5% are used, i.e. more than 95% of the capacity would be available 99% of the time. These resources have been already paid for including their external follow-up costs (environmental etc.). The same holds to less extent for cluster infrastructures due to virtualization strategies. E.g. the Erasmus Computing Grid [11] with ~15,000 PC (~30,000 cores, ~30 Tflops), i.e. a ~22 M€ investment. Thus, in the notoriously under-funded public domain more efficient resource usage by means of grid is of major importance.

In many sectors better resource management can increase the efficiency tremendously. Thus, at least locally the disaster of reaching the (physical) limit can be delayed. A prime example from the production of fundamental raw materials is e.g. the integrated production in the chemical industry [12]: Here byproduct usage, i.e. the *waste* of one process, is reused in another one often even as main process component [13]. Integrated production can reach the level of an extremely fine-tuned ecological organism (as in the highly sophisticated chlorine chemistry) that little changes have severe "survival" consequences for the whole system [14, 15]. In real biological systems, however, there is more flexibility for such changes as in the highly integrated and sophisticated agro-forestry systems e.g. in Indonesia, which have been developed over centuries reaching extremely high efficiencies and are one of the biggest cultural achievements ever [15]. In both cases the efficiency, i.e. the relation between system input and output, are maximized and beat every other process or management [16].

Here, we analyze the internalization challenge of grid technologies [17] into *e-Society* by what we call *e-Human Grid Ecology*, describing adequately the integrated holistic ecology like system parameters and strategies necessary. Therefore, we introduce the novel notion of the *Inverse Tragedy of the Commons*, i.e. that the resources are underused in contrast to their overexploitation. This phenomenon is generic and also found e.g. in the climate issue and is derived from an analysis and combination of the grid sector and the challenges emerging on the micro level of the individual with its security/risk psychology [18] as well as on the macro level of autopoietic social sub-systems [19-21]. To derive points of action, we will extent the classical *Human Ecology* framework [22, 23] to describe the interactions between invironment-individual-society-environment completely. This will then lead to *e-Human Grid Ecology*, a rational allowing to understand and approach the *Inverse Tragedy of the Commons* of grids and other fields with similar paradoxes as those in *(e-)Society*.

2 The Generic Organization of Grid Infrastructures

Since obviously huge resources are available in IT alike the renewable energy sector, despite constant shortage claims, this paradoxical phenomenon can be approached by analysing grid infrastructures and their organization: The Erasmus Computing Grid (ECG) [11] exploits ~15,000 PC (~30,000 cores, ~30 Tflops) donated by the Erasmus Medical Center and the Hogeschool Rotterdam via the middleware CONDOR, a new management system and a central entry port – the ECG Office. It is one of the largest desktop grids for the biomedical research, education and care sectors. MediGRID [24-26] with its services branch Services@MediGRID operate the biomedical research and care grid of the national German D-Grid. The mainly cluster resources are donated and maintained by local universities and connected by different middlewares with local and central access points/portals using high-security protocols under medical conditions. MediGRID is one of the most advanced HealthGrids combining storage, computing and visualizations in the biomedical sectors [24-26].

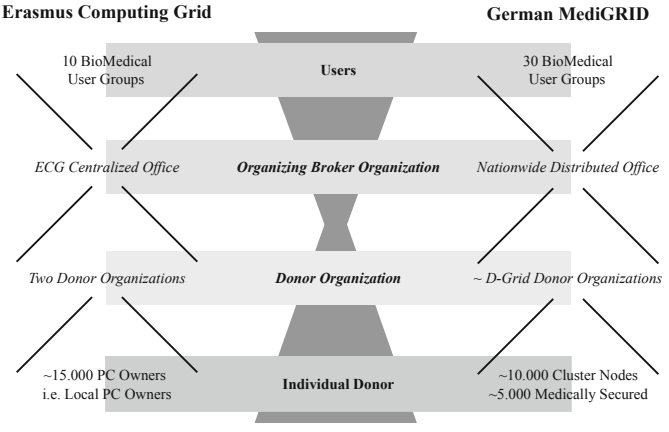


Fig. 1. Abstraction and detailed structure of the Erasmus Computing Grid [11] and the German MediGRID [24-26] showing the four levels involved in grid infrastructures: i) users, ii) organizing broker organizations, iii) donor organizations, and iv) individual donors. Although, the details may vary the structure leads to similar changes on the micro and macro level, which can be understood by the *Human Ecology* rectangle (6.).

Obviously, four levels of organization are involved (Fig. 1) i) users, ii) organizing broker organizations, iii) donor organizations, and iv) individual donors. More abstract this means i) individuals and ii) associations to different degree on each of these four levels. Consequently, there is a micro level from which a macro level emerges, having again an influence on the micro level. The micro level is constituted by an invironment and the macro level creates an environment, which already constitutes the *Human Ecology* rectangle to which we come back later again (6.).

3 The *Inverse Tragedy of the Commons* in Grid Infrastructures

Thus, obviously there are affluent resources and also solutions for their efficient exploitation. Nevertheless, sharing them completely, e.g. by integrated holistic ecology like systems, seems hard since grid infrastructures slowly emerge, which is paradoxical, since IT has great technical opportunities with fast turnover rates.

The analysis of grid organizations showed already that there is a micro level from which a macro level emerges. In the case of over-exploitation of common resources as e.g. in the climate change dilemma, the exactly same complex interplay between the individual and the society as well as the invironment and the environment appears. Here, this is well known as the so called *Classic Tragedy of the Commons* [5-10], in which (multiple) independently acting individuals due to their own self-interest can ultimately destroy a shared limited resource even though it is clear that it is not in the long term interest of the local community and society as a whole. In the case of under exploited resources there seems to be the same phenomenology of the challenge put forward. This until now not theoretically defined phenomenon we - as a logical consequence - name the *Inverse Tragedy of the Commons* and define as (Fig. 2):

The (Classic) Tragedy of the Commons:
A resource belonging to all and being on limited demand is OVEREXPLOITED by the user due to responsibility diffusion!

<=> *TRANSFORMATION* <=>

:The INVERSE Tragedy of the Commons
A resource belonging to all and being in affluent availability on limited demand is UNDEREXPLOITED by potential users due to responsibility diffusion!

Interestingly, not only is responsibility diffusion the most likely and general reason for the appearance of both tragedies, but also the psychological description for both the micro and the macro level hits the same archetypical traits (Fig. 2). The under-used potentials and over-used resources clearly show how from a virgin resource opportunity, concrete objects emerge with their attached limitation burden (Fig. 2). Thus, the complex field created, describes exactly the tension found in resource limitation phenomena and evolutionary emergence.

Pinpointing the phenomenon of under exploited potentials to the same phenomenological root as the well known phenomenon of over exploited resources opens now the complete opportunities and tool set to examine the challenge of introducing grid as well as its principle day-to-day management.

4 The Grid *Tragedy of Autopoietic Social Sub-systems*

The challenge of integrating resources in a virtualized manner involves naturally all stakeholders of society (Fig. 1). The existence of a grid *Inverse Tragedy of the Commons* and its macro social aspects, point to the major importance of the interaction complexity of the social sub-systems theory by Niklas Luhmann [20, 21]. It is based on the autopoietic concept of Humberto Maturana and Francisco Varela [19] - the most advanced social systems theory, describing the huge complexity of the macro sociality of the grid phenomenon. An autopoietic system is a network of processes consisting of: i) interactions and transformations continuously regenerating and realizing its networks of existence, and ii) the constitution of the system as a unity in space in which the component exist by specifying the topological domain of its realization. Central to this description of evolutionary emergence, i.e. self-reproducing systems, is the material and information exchange between the components. Social systems are obviously communication systems, with society being the most encompassing one. Consequently, many of the conundrums appearing during the society internalization become evident and are in agreement with the *Inverse Tragedy of the Grid Commons*:

In detail around seven social sub-systems can be defined and reflect the evolutionary emergence from deep psychology to society [21]: i) religion, ii) education, iii) science, iv) art, v) economy, vi) jurisdiction, vii) policy. All of these systems have their internal code of communication and their own connectivity interface to the other sub-systems. This results in hug barriers: e.g. the religious code of believe or not-believe is incompatible with the have or not have money code of the economic sector. This is even more true for science (true vs. non-true), jurisdiction (just vs. un-just) and politics (power vs. no-power), which have nothing to do with education (knowledge vs. no-knowledge). Since grid infrastructures belong currently mostly to the academic

sector [11, 24-26], their widespread usage within society, i.e. their internalization into society, is deaccelerated by the lack of interoperability between these sub-systems. Consequently, the *Inverse Tragedy of the Commons* results in

The Tragedy of Autopoietic Social Sub-Systems:

Sub-systems have their own code of communication and are separated from each other in a way blocking in principle a consistent integration although they form a society with all their contradictions this leads to blockage of the system.

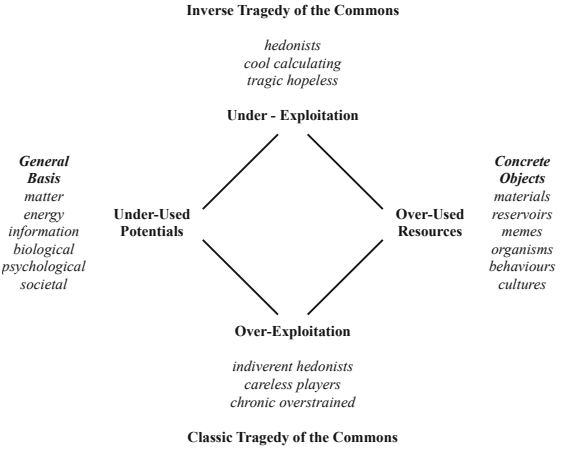


Fig. 2. Generalization of the *Inverse Tragedy of the Commons*: the *Classic* and *Inverse Tragedy of the Commons* are directly complementary to each other and can on the deep psychology level associated with complementary pears of archetypical behavioural traits. This is in line with the complementary under-used potentials and over-used resources, which emerge from the potentials by freezing of potentials into concrete objects with corresponding limits.

This macro level tragedy clarifies that grid organizations are just another example for complex infrastructures whose efficiency increase depends beyond more or less complex technical solutions on the participation of all sub-systems concerning their societal internalization. This analysis and its acceptance is an important knowledge opening huge opportunities to examine and approach the challenge of introducing grids and their management. Beyond, this analysis makes the challenges in other sectors, which might reside in other sub-systems, obvious as well. Consequently, the *Classic* and *Inverse Tragedy of the Commons* can be understood as a societal challenge with the opportunity to be resolved.

5 The *Grid Tragedy of Security/Risk/Profit Psychology*

Since the macro level of social sub-systems emerges evolutionarily from the micro level [18], one needs to consider the individual for whom each implementation and internalization of a new technology is based on a positive relation between the risk and the profit involved. Thus, the level of altruism leading to successful sharing on

the individual level and its commitment beyond its own job/agenda, as well as that of its own institution without incentive structure to take responsibilities, also leads to responsibility diffusion. Even the clear win-win situations for individual grid users are under these circumstances hard to communicate and even the additional networking effects result hardly in the set-up or usage of grids. It is also unlikely that people take the risk to exceed their own budget and corresponding responsibilities, when future results and its benefits are unclear to them. As long as sharing is voluntary and in hand with uncertainty and risks, it is less likely that individual problem owners will behave altruistically on behalf the societal benefit. Consequently, on the micro level the situation is that of a perverse *Inverse Tragedy of the Commons*: the commons is not abused or overexploited, but in contrast the tremendous resources are not used at all despite the needs and obvious benefits, due to secondary irrational interests.

Thus, the integration challenge involves the individual of the different institutionalized society stakeholders. These individuals shape the individual actions according to their function in a social sub-system. How an individual perceives the security/risk/profit ratio depends on its personal security/risk/profit psychology matrix:

Deep Psychology Security/Risk/Profit Cascade:	*:Autopoietic Sub-System Correspondence*
emotional individual s/r/p perception	genetics and deep psychology
rational s/r/p knowledge acceptance	education and science
internalized incidental s/r/p behaviour	economics
accepted legal and political s/r/p scenarios	jurisdiction and politics
lived religious and cultural s/r/p archetypi	religion, art and culture

Thus, this matrix describes a similar challenge on the micro level similar to the macro level with conflicting personal positions and internal balancing the invironment with the environment. This creates on the micro level again a tragedy:

The Tragedy of Security/Risk/Profit Psychology:

Individuals balance constantly a complex combination of invironmental and environmental security/risk/profit deep psychology factors whose contradictions lead to responsibility diffusion.

Unfortunately, the identification and analysis of this tragedy is by far more challenging concerning management guidelines, due to the hardly changeable basis (e.g. the genetic basis) and the time scales involved, in contrast to the macro level, where bypassing measures and changes can in principle be implemented *at will*. Consequently, this tragedy has to be tackled with big care and shows that the *Inverse Tragedy of the Commons* can really be addressed by a *Human Ecology* rectangle approach (6.).

6 *e-Human Grid Ecology*

To overcome the "dare-to-share" attitude in respect to the *Inverse Tragedy of the Commons* and its base tragedies in the grid sector, a sustainable grid approach in an

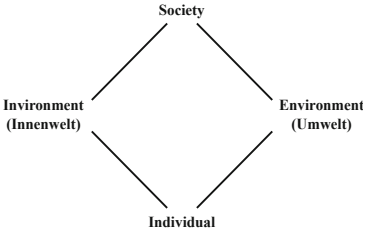


Fig. 3. The *Human Ecology* rectangle describes the relation between the invironment (Innenwelt), the individual, the society and the environment (Umwelt). It is the extension of the incomplete classical *Human Ecology* triangle, which consists only of the individual, the society and the environment but not the invironment. The invironment and the environment as well as the individual and the society are complementary pairs spanning a field between them. The invironment thereby constitutes the Innenwelt; the individual forms society; to the individual society gets a general environment as the invironment constitutes much of the society.

ecology-like manner combining the micro and macro level is crucial. Since these are similar phenomena as e.g. in climate protection, the interdisciplinary applied field of *Human Ecology* [15, 18, 22, 23] gives a framework for their combination, followed by understanding and approaching direct guidelines of the management of grids. *Human Ecology* was developed originally by Robert Park (1864-1944) and Ernest Burgess (1886-1966) and evolved in Chicago in the 1920's in close connection to the field of city development. *Human Ecology* classically deals with the complex interplay between i) the individual, ii) the society, and iii) the environment, which usually is illustrated by the so called *Human Ecology* triangle. This framework is used to investigate many a complex mankind related challenges as e.g. the exponential demand growth until reaching a limit, its inherent property of life and evolution, as well as waste and pollution related issues. Obviously, these sustainability questions beyond the materialistic world are found on all evolutionary levels up to the psychological, societal and cultural one and involve also every cause for exponential growth.

The detailed analysis of the *Inverse Tragedy of the Commons* by investigating the *Tragedy of Autopoietic Social Sub-Systems* and the *Tragedy of S/R/P Psychology*, proposes the extension of the classical *Human Ecology* triangle to a rectangle consisting of: i) invironment ii) individual, iii) society, and iv) environment (Fig. 3). It describes the relation between the invironment (Innenwelt), the individual, the society and the environment (Umwelt). The invironment and the environment as well as the individual and the society are complementary pairs and create a field. The invironment thereby constitutes the Innenwelt, the individual forms society, the individual society creates an environment as the invironment constitutes much of the society. The rectangle reflects the mirco level (invironment and individual) and the macro level (society and environment) correctly. This fits the field of the *Classic* and *Inverse Tragedy of the Commons*, with its under-used potentials and over-used resources.

Consequently, the classic *Human Ecology* paradigms to understand the *Classic* and *Inverse Tragedy of the Commons* can now be used to define the corresponding subfield for *e-Live* and *e-Society* with *e-Invironment* and *e-Environment* extending the

classic notion of ecology and using it in its detailed consequences for the understanding and management to internalize grids into society to increase dramatically the efficiency of resource exploitation dramatically to its e-ecology sustainable limits.

The Definition of e-Human "Grid" Ecology

*"Under **e-Human "Grid"** Ecology we understand the complete science of the relationships of **grid** to the surrounding environment to which we can count all conditions of existence in the widest sense." [27]*

***(e-Human "Grid"** Ecology" is)...*
*the relationship between **grid** and all other **e-Social sub-systems**." [28]*

7 Conclusions

It is obvious that there are huge resources available in IT as in most other production and social sectors resulting in many opportunities. Similar phenomena are known for many sectors as e.g. the renewable energy sector. In contrast, ever more resources are said to be required but believed to be at their limit and thus already unavailable for further exploitation. Especially in IT the demand for information storage and processing in R&D grows exponentially. Although exponential growth inevitably will lead to the reaching of limits sooner or later, there seems to be also many an opportunity to sustainably manage resources on very long time scales. Thus, clever resource management can increase the efficiency tremendously and in consequence avoid limiting barriers as e.g. integrated chemical production or sophisticated agro-forestry systems show. Grid infrastructures in IT are believed to be another of these solutions exploiting under-used and available resources. The IT sector and there especially the grid area with their fast technological turnover rates would, however, allow to bring innovation opportunities fast to the market and thereby increase the efficiency of computing and storage usage tremendously. This could result on the one hand in lower investments into infrastructure, which obviously would provoke large resistance by the producing industry, or on the other hand would result in a much higher output and thus return of investment made by society in these infrastructures, which would give a big “present” with only minor further investment to society as a whole. Nevertheless, it remains a big issue despite all the efforts of the grid and other communities, why the obvious huge benefits of much higher resource exploitation efficiencies is so hard to internalize into societies despite its crystal clear benefits.

The organizational architecture analysis of grid infrastructures and their management shows that there are four levels of stake-holders involved: i) users, ii) organizing broker organizations, iii) donor organizations, and iv) individual donors. That is a much more complicated situation compared to that of the integrated production in chemical industry, since here we have a large spatial and cultural coverage in contrast to the “internal” situation of one single company. There is also a big difference to sophisticated agro-forestry systems as e.g. those in Indonesia, since these systems had a huge temporal time span for development. Although they involve in principle the entire society, the decisions are still taken by the single farmer and community despite their *a posteriori* internalization in tradition and cultural rules. Abstraction of the four levels involved in grid infrastructures leads to a micro level from which a macro level

emerges, having again an influence on the micro level and *vice versa*. The micro level is constituted by an invironment and the macro level creates an environment, which already constitutes the *Human Ecology* rectangle as we showed later. Consequently, here from the pure theoretical viewpoint we have not only reached complete consistency proving the validity of our arguments, but also gained access to a "tool box" used successfully for complex internalization issues. This is important for generalization and for justification thereof derived management measures.

The fact that IT resources are completely underused we identified additionally and for the first time as a phenomenon of the *Inverse Tragedy of the Commons*, i.e. that the resources are not overexploited unsustainably until their destruction. Together with the finding of the micro and macro level in the organization of grid organizations which plays the important role in both the *Classic* and *Inverse Tragedy of the Commons* leading to responsibility diffusion and thus inefficient resource management, consequently makes clear that the grid challenge concerning implementation and integration of grids lies in the social embedding of the micro and macro level phenomena: i) the sharing attitude/socialisation based on the security/risk/profit psychology of the individual, and ii) the culture of the embedding institution and society based on the interaction of the autopoietic social sub-systems. This is similar e.g. to the renewables sector and thus has huge implications beyond the grid sector.

Considering the macro level more in detail reveals that the autopoietic sub-systems theory describes the social environment best. Unfortunately, the social sub-systems i) religion, ii) education, iii) science, iv) art, v) economy, vi) jurisdiction, vii) policy, have a more or less incompatible code of communication which leads to the *Tragedy of the Autopoietic Social Sub-Systems* leading to large inconsistencies and blockings. Consequently, the grid challenge lies in the integration of autopoietic sub-systems towards a working grid society on the micro and macro level by i) approaching the sub-system stickyness of individuals, and ii) the soft bridging of sub-systems. In the daily work that means that the individuals have first to realize their own working and borders of their social sub-system as well as that of the other social sub-systems. In a second step the possible bridges between social sub-systems need to be realized and concrete ways to circumvent inherent blockings have to be explored to actually get to a working solution to reach the level of a joined effort to realize a project.

On the micro level the security/risk/profit psychology matrix plays the major role since for the individual each internalization of a new technology is based on a positive relation between the security/risk/profit involved. Even for clear win-win situations the phenomenon of responsibility diffusion can appear. Since individuals have to balance constantly between the invironment and the environment, i.e. between psychology and social sub-systems, there appears also a hard to tackle *Tragedy of the Security/Risk/Profit Psychology*. Consequently, the grid challenge on the micro and macro level are given by i) the individual perception and the individual well being, and ii) the procedural and institutionalized careful management. For the daily work that means that the individual need to rationalize its own behavioural background and invironmental constituency, and that institutions need to accept and develop the invironment of their employees as well as the psychological status of the environment they create. Thus, the creation of awareness might not change the individual but by team formation with different characters and corresponding procedures, the openness in an institutionalized form can increase the internalization of new technologies.

Consequently, to overcome the "dare-to-share" attitude in respect to the *Inverse Tragedy of the Commons* and its base tragedies, the *Tragedy of Autopoietic Social Sub-Systems* and the *Tragedy of Security/Risk/Profit Psychology* in the grid sector, a sustainable grid approach in an ecology-like manner combining the micro and macro level is crucial for the grid sector. The interdisciplinary field of *Human Ecology* gives a framework also for the understanding and approaching of the *Classic* and *Inverse Tragedy of the Commons* for direct guidelines in the day-to-day management of grids as well as other areas and combine the statements made above into a unified framework. Theoretically our analysis showed that it is necessary to extent the classical *Human Ecology* triangle to rectangle with i) the invironment ii) the individual, iii) the society, and iv) the environment. As mentioned before from the pure theoretical viewpoint, we have not only reached complete consistency proving the validity of our arguments, but also gained access to a “tool box” which has been used already successfully. This is important for generalization as well as for justification of thereof derived management measures. To rationalize that further on the theoretical level, this can be used to define the corresponding notion for *e-Live* and *e-Society* with *e-Invironment* as well as *e-Environment* extending the classic notion of ecology, which leads to *e-Human Grid Ecology*. Consequently, the *Inverse Tragedy of the Grid Commons* can be tackled on the micro and macro level: i) participative integration of the individual and acceptance of its invironment, and ii) sustainable integration of e-Social autopoietic sub-systems and its environment. Thus, we propose an ecological systems framework to deal with the phenomenon of under-used resources in the IT sector similar to the paradigms of integrated production in chemical industry and sophisticated agro-forestry systems e.g. in Indonesia.

In summary, we think that with the extended *e-Human (Grid) Ecology* rational, the *Inverse Tragedy of the Commons* of the grid and other sectors as e.g. the renewable energy field, cannot only be understood but also approached concerning the detailed daily work of implementation and internalization of new efficient strategies into society, which is of major importance for many fields where similar paradoxes appear and thus finally for the well-being of mankind and planet earth in general.

Acknowledgements

We are very thankful to the German and International Societies for Human Ecology and their members for discussion of the background of this work. We also would like to thank Nick Kepper, the Erasmus Computing Grid as well as the German MediGRID and Services@MediGRID consortia for their support. This work was partly supported by the Erasmus Medical Center and the Hogeschool Rotterdam, Rotterdam, The Netherlands, as well as the Bundesministerium für Bildung und Forschung (BMBF) under grant # 01 AK 803 A-H (German MediGRID) and # 01 IG 07015 G (German Services@MediGRID) to TAK.

References

1. Faber, M., Manstetten, R.: Mensch – Natur – Wissen: Grundlagen der Umweltbildung. Vandenhoeck and Ruprecht (2003), ISBN 3525301413, ISBN 978-3525301418
2. Egger, K., Glaeser, B.: Ideologiekritik der Grünen Revolution: Wege zur technologischen Alternative, Technologie und Politik. Rororo Actuell, Reinbeck (1975)
3. Faber, M., Niemes, H., Stephan, G., Freytag, L.: Entropy, Environment, and Resources. An Essay in Physico- Economics. Springer, Heidelberg (1987), ISBN 3540182489, ISBN 978-3540182481
4. IPCC, `http://www.ipcc.ch` and `http://www.ipcc.ch/ipccreports/index.htm`
5. Hardin, G.: Tragedy of the Commons. Science 162, 1243–1248 (1968)
6. Ostrom, E.: Governing the commons: The evolution of institutions for collective action. Cambridge University Press, Cambridge (1990), ISBN 0521405998, ISBN 978-0521405997
7. Hardin, G.: The Tragedy of the Unmanaged Commons. Trends in Ecology & Evolution 9, 199 (1994)
8. Hardin, G.: Extensions of "The Tragedy of the Commons". Science 280, 682–683 (1998)
9. International Association for the Study of the Commons (IASC), `http://www.iascp.org`
10. Tragedy of the commons, `http://en.wikipedia.org/wiki/Tragedy_of_the_commons`
11. de Zeeuw, L.V., Knoch, T.A., van den Berg, J., Grosveld, F.G.: Erasmus Computing Grid - Het bouwen van een 20 TeraFLOP virtuelle supercomputer. In: Frederik, H. (ed.) NIOC proceedings 2007 - het perspective of lange termijn, NIOC, Amsterdam, The Netherlands, pp. 52–59 (2007)
12. Faber, M., Stephan, G., Michaelis, P.: Umdenken in der Abfallwirtschaft. Vermeiden, Verwerten, Beseitigen. Springer, Heidelberg (1989), ISBN 3540518398, ISBN 978-3540518396
13. Jentzsch, W.I.: BASF Antwerp – an integrated chemical plant. Industrial Lubrication and Tribology 47, 4–5 (1995)
14. Faber, M., Schiller, J.: Joint Production and Responsibility in Ecological Economics: On the Foundations of Environmental Policy (Advances in Ecological Economics). Edward Elgar Publishing (2006), ISBN 1840648724, ISBN 978-1840648720
15. Egger, K., Rudolph, S.: Zum anschaulichen Umgang mit komplexen Aspekten der Kultur- und Ökokrise. In: Glaeser, B., Teherani-Krönner, P. (eds.) Humanökologie und Kulturökologie – Grundlagen, Ansätze, Praxis, pp. 191–220. Westdeutscher Verlag, Opladen (1992)
16. Faber, M., Manstetten, R., Proops, J.: Ecological Economics: Concepts and Methods. Edward Elgar Publishing (1998), ISBN-10: 1858989981, ISBN 978-1858989983
17. Foster, I., Kesselman, C.: The grid: blueprint for a new computing infrastructure. The Elsevier Series in Grid Computing. Morgan Kaufmann, Amsterdam (2004), ISBN 1558609334
18. Egger, K.: Die moralische Inversion – Ursachen unserer normativen Orientierungskrise und deren Folgen für die Agrarentwicklung – Alternativpositionen durch Dialog mit anderen Kulturen. In: Bruckmeier, K., Serbser, W.H. (eds.) Ethik und Umweltpolitik: Humanökologische Positionen und Perspektiven, Oekom Verlag (2008), ISBN 978-3-86581-119-6
19. Maturana, H.R., Varela, F.: Tree of Knowledge. Shambhala (1992), ISBN 0877736421, ISBN 978-0877736424

20. Luhmann, N.: Ökologische Kommunikation. VS Verlag für Sozialwissenschaften (2004), ISBN 3531517759, ISBN 978-3531517759
21. Luhmann, N.: Soziale Systeme: Grundriß einer allgemeinen Theorie. Suhrkamp (2008), ISBN 3518282662, ISBN 978-3518282663
22. Egger, K.: Evolution, Menschenbild und Umweltkrise – ein Versuch zur humanökologischen Hypothesenbildung. In: Wehrt, H. (ed.) Humanökologie – Beiträge zum ganzheitlichen Verständnis unserer geschichtlichen Lebenswelt, pp. 155–189. Birkhäuser, Basel (1996)
23. Bruckmeier, K., Serbser, W.H. (eds.): Ethik und Umweltpolitik: Humanökologische Positionen und Perspektiven. Oekom Verlag (2008), ISBN 978-3-86581-119-6
24. Sax, U., Weisbecker, A., Falkner, J., Viezens, F., Yassene, M., Hartung, M., Bart, J., Krefting, D., Knoch, T.A., Semler, S.C.: Grid-basierte Services für die elektronische Patientenakte der Zukunft. E-HEALTH-COM - Magazin für Gesundheitstelematik und Telemedizin 4, 61–63 (2007)
25. Sax, U., Weisbecker, A., Falkner, J., Viezens, F., Mohammed, Y., Hartung, M., Bart, J., Krefting, D., Knoch, T.A., Semler, S.C.: Auf dem Weg zur individualisierten Medizin - Grid-basierte Services für die EPA der Zukunft. In: Jäckel, A. (ed.) J. Telemedizinführer Deutschland 2008, pp. 47–51. Deutsches Medizinforum, Minerva KG, Darmstadt (2008), ISBN 3-937948-06-6, ISBN-13 9783937948065
26. Krefting, D., Bart, J., Beronov, K., Dzhimova, O., Falkner, J., Hartung, M., Hoheisel, A., Knoch, T.A., Lingner, T., Mohammed, U., Peter, K., Rahm, E., Sax, U., Sommerfeld, D., Steinke, T., Tolxdorff, T., Vossberg, M., Viezens, F., Weisbecker, A.: MediGRID - Towards a user friendly secured grid infrastructure. Future Generation Computer Systems 25, 326–336 (2008)
27. Haeckel, E.: Generelle Morphology der Organismen. Band 2 Allgemeine Entwicklungsgeschichte. Berlin, p. 286 (1866)
28. Haeckel, E.: Natürliche Schöpfungsgeschichte, 9. Auflage, Berlin, p. 793 (1898)

Towards a Generic Value Network for Cloud Computing

Markus Böhm[1], Galina Koleva[1], Stefanie Leimeister[2], Christoph Riedl[1], and Helmut Krcmar[1]

[1] Technische Universität München (TUM), Chair for Information Systems
[2] fortiss – Research Institute at Technische Universität München (TUM)
Boltzmannstr. 3, 85748 Garching b. München, Germany
{markus.boehm,galina.koleva,
stefanie.leimeister,riedlc,krcmar}@in.tum.de
http://www.winfobase.de

Abstract. With the rise of a ubiquitous provision of computing resources over the past years, cloud computing has been established as a prominent research topic. In contrast to many other research works, this paper does not focus on technical aspects of cloud computing but rather takes a business perspective. By taking this perspective we examine the ecosystem that has developed around cloud computing. Here, new market players emerged, breaking up the traditional value chain of IT service provision. In this paper we describe the roles of different market actors and develop a generic value network of cloud computing, using the e³-value method. Based on interviews with domain experts we were able to draw first estimates regarding possible future value streams within the ecosystem. Extending the prevailing technical perspective of cloud computing, this paper shifts the focus to a broader understanding of business opportunities and business value. Researchers can apply the developed generic value network as an analytical framework to guide their research, while practitioners might apply it to position themselves in the cloud computing market and identify possible business opportunities.

Keywords: Cloud Computing, Market Actors, Value Network, Value Chain.

1 Introduction

With the rise of a ubiquitous provision of computing resources over the past years, cloud computing has been established as a prominent research topic. It can be seen as an innovation in different ways. From a technological perspective it is an evolution of computing that evolved from large tabulating machines and mainframe architectures, centrally offering calculating resources to personal computers for decentralized computation, and eventually to ubiquitous, small personal (handheld) devices [1].

While much research is dedicated to the technical aspects of cloud computing, many authors neglect the business perspective of IT provisioning. From this perspective cloud computing has the potential to revolutionize the mode of computing resource and application deployment, breaking up traditional value chains and making room for new business models. Many providers like Amazon, Google, IBM, Microsoft, Salesforce.com, or Sun position themselves as platform and infrastructure providers in the

J. Altmann and O.F. Rana (Eds.): GECON 2010, LNCS 6296, pp. 129–140, 2010.

cloud computing market. Alongside, there emerge other providers, who build their own applications or consulting services upon the services offered by the before mentioned. This eventually leads to a whole new ecosystem of service providers in the cloud computing market.

So far, only little research has been conducted around value creation, value chains and value networks in cloud computing. Some studies have been conducted in the context of grid computing. Stanoevska-Slabeva et al. [2] describe different grid stakeholders and have developed a generic value chain and a corresponding value network for grid computing based on an analysis of industry case studies. Another work by Altmann et al. [3] developed a taxonomy and description of stakeholders and their roles in grid computing. Lee and Leem [4] have studied the value chain for a ubiquitous computing environment. A value chain reference model explicitly developed for cloud computing was presented by Mohammed et al. [5], based on Porter's value chain theory [6]. Their work, so far, appears to be the most comprehensive value chain reference model for cloud computing. They distinguish between primary (core) services, business oriented support services and cloud oriented support services. Primary services include hardware, grid middleware, software and data & content services. Business oriented support services include resellers, composers, financial services and market places, while cloud oriented support services are comprised of technology operators, grid financial management services, solution and consultant services as well as customized services.

However, Mohammed et al.'s [5]work is settled on a comparatively micro-economic level, describing service scenarios, including costs and profits as well as serving as a check-list for building cloud services. We could not identify any detailed work around the cloud computing ecosystem, which takes a comparatively macro-economic perspective on cloud computing. Our objective is to describe this emerging ecosystem with its market actors and their value exchange relationship. To achieve this, our research was guided by the following two questions:

a) What generic market actors can be identified in the cloud computing market?

b) How does the ecosystem of market actors and value exchanges look like?

2 Cloud Computing Definition

Until today no common definition of cloud computing has emerged. Youseff et al. were among the first who tried to provide a comprehensive understanding of cloud computing and all its relevant components. According to them "cloud computing can be considered a new computing paradigm that allows users to temporary utilize computing infrastructure over the network, supplied as a service by the cloud-provider at possibly one or more levels of abstraction" [7]. By levels of abstraction, the authors distinguish between cloud applications (Software as a Service, SaaS), cloud software environment (Platform as a Service, PaaS) and software infrastructure (Infrastructure as a Service, IaaS) based on a software kernel and hardware. Despite different definitions (an overview can be found in [8]), there is some consent, which can be summarized as the dynamic, on demand provision of services via a network which are priced according usage. In our understanding, "Cloud Computing is an IT deployment

model, based on virtualization, where resources, in terms of infrastructure, applications and data are deployed via the internet as a distributed service by one or several service providers. These services are scalable on demand and can be priced on a pay-per-use basis"[8]. This definition was derived on the basis of a review of scientific definitions, taking a holistic view of cloud computing from applications to infrastructure, stressing the ability of service composition. Thus it supports our business oriented perspective and our observation of a growing ecosystem of different market actors around cloud computing.

3 Value Systems Concepts

3.1 Value Chain

The value chain is a model that describes a series of value-adding activities connecting a company's supply side (raw materials, inbound logistics, and production processes) with its demand side (outbound logistics, marketing, and sales). By analyzing these activities, managers are able to redesign their internal and external processes to improve efficiency and effectiveness and to identify their core competencies [9]. As such it is often used to analyze a firm and its major competitors by identifying differences in performance (benchmark) [10].

The most established value chain approach was presented by Porter, who distinguishes between primary activities (inbound logistics, operations, outbound logistics, marketing & sales, services) and support activities (firm infrastructure, human resource management, technology development, procurement), also adding a value margin [6]. To account for the collaboration between companies, Porter created an extended value chain, termed *value system*. It represents an interconnected system of value chains. A value system includes the value chains of a firm's supplier (and their suppliers all the way back), the firm itself, the firms distribution channels, the firms customers and so forth to the end customer (consumer). Linkages connect value activities inside a company but also create interdependencies between the members of the value system [6]. A company can create competitive advantage by optimizing or coordinating these links to the outside [11].

3.2 Value Network

The value network focuses on value co-creation by a combination of actors within a network. A value network is a "set of relatively autonomous units that can be managed independently, but operate together in a framework of common principles and service level agreements (SLAs)" [10]. Within such a network, value for the consumer is created at the network level, where each actor contributes incremental value to the overall offering [12]. Instead of providing the maximum value to the customer, which is always at risk of being unprofitable, actors concentrate on their core competencies and the competence complementarity of the network [13, 14].

Biem and Caswell have examined and compared several definitions of the value network. Their derived definition puts a special emphasis on the increase of inter-firm relationships. A "value network is a model of inter-organizational exchange as an

attempt to address the increasing intricateness of inter-firm relationships, pushed by a more and more connected economy" [15].

3.3 Critical Comparison

Sturgeon [9] analyzed the key differences of value chains and value networks. Both approaches have many things in common but a fundamental distinction can be made. While the value chain "maps the vertical sequence of events leading to the delivery, consumption, and maintenance of a particular good or service, the value network maps both the vertical and horizontal linkages between economic actors, i.e., recognizing that various value chains often share common economic actors and are dynamic in that they are reused and reconfigured on an ongoing basis" [9].

Another differentiation can be seen by the order activities are carried out. The value chain is a linear approach, which perfectly represents a manufacturing process in traditional industries, following a sequential logic. The value network on the other hand is a model, where functions and activities are performed simultaneously rather than sequentially [10, 17]. It can better display alliances and cooperation relationships. Firm relationships have increased in complexity. Firms can no longer be simply classified as customers, suppliers or competitors. Often they have two or more of these dimensions simultaneously [15, 17].

Yet another aspect is the dematerialization and digitization of products and the trend towards service delivery. According to Peppard and Rylander [10] every business today competes in two worlds: a physical world of resources that can be seen and touched and a virtual world made of information. The latter has given rise to the world of electronic commerce, a new way of value creation. Processes in creating value are not the same in both. The value chain treats information as a supporting element in the value-adding process, but it can also be a source of value in itself. Therefore, the value chain is better suited to present the physical value creation while the value network is more suitable for the virtual world. The latter is particularly evident in sectors such as banking, insurance, telecommunications, news, entertainment, music, advertising, and of course cloud computing.

As this discussion highlights, value networks appear to suit better for modeling the interdependencies between different actors observed in the cloud computing market.

4 Methodology

By analyzing literature on market actors in the areas of living systems theory [18], network organizations [19, 20], (electronic) business webs [21-25], IT outsourcing [26], grid computing [2, 3], and value networks in traditional industries [27, 28] we could identify 105 generic roles. After clustering these roles based on their descriptions provided by the authors we reflected them with the observed cloud computing market. Therefore we have built a dataset of 2628 cloud computing services based on convenient sampling. The services from the dataset were then assigned to actor clusters and it has been checked, whether some services could not be assigned to any service. At the end of this process we ended up with a set of 8 generic roles described in section 5.1.

We developed our conceptual generic value network, based on the roles described in section 5.1 and modeled it with the e³-value method by Gordijn and Akkermans [29]. This value network, explained in section 5.2, was then evaluated through ten interviews with domain experts. The interviews, lasting between 45 and 90 minutes, were conducted between October and December 2009, following a semi-structured interview guideline. The interviewees were selected based on their expertise in the cloud computing area, demonstrated by their active participation in BITKOM's[1] cloud computing task force. To improve reliability, the interviews were tape-recorded and transcribed. The semi-structured interviews covered the experience of the interview partner with cloud computing, different roles and their relationship amongst each other as observed by the interviewee, and the presentation and validation of our proposed value network. The interviewees were also asked for their estimation of the value creation and flow within the value network based on three distinct scenarios.

5 A Generic Value Network of Cloud Computing

5.1 Generic Roles and Market Actors

Due to an increasing trend towards service orientation, opportunities to offer services on cloud computing platforms, and the possibilities to integrate individual component services to create value-added, complex services gave rise to a set of new roles in cloud computing and the resulting service ecosystems [30, 31].

Cloud computing services are typically classified by the type of service being offered. With reference to Youseff et al.'s cloud computing ontology [7], cloud services are often differentiated by application (SaaS), platform (PaaS) and infrastructure (IaaS) level. In contrast to this layer model, which is quite common in the IT domain, cloud services can also be classified in a more business-oriented manner, by market actors that offer a certain class of services. Since market actors represent companies that might offer different services on different levels, such as Salesforce.com, we abstract from this construct and speak of roles. In our understanding a role is a set of similar services offered by market players to similar customers. This abstraction helps us to indicate that a certain company (market player) can offer different services, acting in different roles. The generic roles described below were derived by clustering roles identified in the literature discussed above and a proceeding reflection in practice.

Application Provider
The application provider offers applications for its customers. In contrast to the traditional software model the applications are hosted and operated by the application provider in an own or outsourced datacenter and are accessible for customers via the internet. The application provider has to ensure a smooth operation of the applications. This includes monitoring, asset/resource management and failure/problem management. Monitoring means that the service provider is aware of the state of his system at any time. Asset/resource management aims to maximize datacenter utilization

[1] BITKOM is the federal association for Information Technology, Telecommunications and New Media in Germany (www.bitkom.org).

by, for example, load balancing [32]. Failure/Problem management refers to both, instant fixes of problems such as bugs as well as to long term software maintenance to avoid problems in advance. Also new features and further application improvements are provided and installed [33]. Yet another important aspect is the security of the software. Unwanted attempts of accessing or manipulating the software have to be detected and stopped (intrusion detection) [34].

(Technical) Platform Provider
We distinguish between two kinds of platform providers. One is the more technically oriented platform provider described here; the other is a market platform, described below. (Technical) platform providers offer an environment to develop, run and test applications. From a technical perspective an operating environment, application programming interfaces (APIs), programming languages etc. are provided. Furthermore, team collaboration services may be offered [35]. Developers are shielded from technical, infrastructure related details. Programs are executed over datacenters, not concerning the developers with matters of resource allocation. This however comes at the cost of some trade-offs and development constraints, possibly leading to a different application design. For instance, depending on the platform, key-value stores have to be used instead of relational databases [36].

Market Platform
The market platform represents a marketplace where various cloud computing services of different roles are offered. The main objective of the market platform is to bring customers and service providers together. The former can search for suitable cloud computing services while the latter can advertise its services. In addition to offering a platform for marketing and searching services, the market platform might also offer additional services to both service providers and customers, such as SLA contracting or billing.

Infrastructure Provider
The infrastructure provider offers virtual hardware, network connections including firewalls and virtual storage to its customers. The customer has full responsibility for the received machine instances and controls them. Once a machine reaches its performance limits, another machine has to be instantiated manually to scale the application [36]. As cloud computing matures, more and more infrastructure providers offer potentially different SLAs to their customers, regarding for example availability and performance [37]. Disaster recovery for both, infrastructure and data is an important aspect of the infrastructure provider's work. [33].

Consultant
To introduce a cloud computing project in a company consultants are often asked for their expertise. Consultants can provide fundamental knowledge about cloud computing offerings as well as the customer company‘s business processes and requirements to identify and introduce suitable cloud services. Additional services might be a cost benefit evaluation to decide whether cloud computing is profitable or not, security consulting or billing. Consulting services are not limited to users of cloud services, but may also target service providers to solve technical problems, evaluate the service offering or analyze customers.

Aggregator
With cloud computing a large number of small and modular services arose, creating the opportunity to aggregate these services into value-added, complex solutions for certain needs. This aggregation of services is accomplished by aggregators. According to the market analyst Gartner three different types of aggregators may arise within the cloud computing context: The first one combines existing services, created by different providers into a new service. The aggregator has to ensure that the different services work together neatly and that no losses occur via data movement between the systems. The second type of aggregator is comparable to a value added reseller. He adds value on top of a given service to ensure some specific capability. These might be add-ons or new services. The last type may categorize and compare cloud services from different providers, based on certain selection criteria. The end user can specify its criteria and get the best fitting solution for its needs [38]. The latter type might also fit into our generic role of the market platform.

Integrator
Once a company decides to integrate a cloud computing solution, the system integrator faces two main tasks. The first is to convert preexisting on-premise data in order to migrate it into the cloud or prepare it for certain applications. The second task is to integrate a cloud computing solution into the existing IT landscape, developing interfaces to other on-premise applications. This also includes system and integration tests to ensure a seamless cooperation of different systems as well as the training of users. Beyond the integration project, the integrator might also offer additional training or customer support, by setting up a help desk for example [39]. There might be some similarities between the aggregator and the integrator role in terms of aggregating different modular services into a more complex solution. The main difference between these two roles is that the integrator creates a customer individual solution, whereas the aggregator develops a more standardized solution that is offered to a larger group of users with similar needs.

Consumer
The consumer is the final customer who receives services for business or private use. He does not create value within the cloud computing ecosystem, nor does he offer cloud computing services to someone else. The consumer is the starting point of service request and the ending point of service delivery. All value adding activities are eventually paid by the consumer.

5.2 A Generic Value Network

Based on our understanding of the different roles emergent in the cloud computing ecosystem, we are able to develop a generic value network, using the e³-value method to model it. Figure 1 depicts the different roles, their interrelationships and value exchanges. Within this value network value is created by providing services that are valuable for other participants of the network. Products or in our case services are exchanged in return of either money, which is the typical case, or other benefits that the service provider values. Value is created by producing elementary services

(infrastructure) and refining them throughout the value network. In this way value is added with each step along a path in the value network until the *Consumer* receives the service that fulfils his needs.

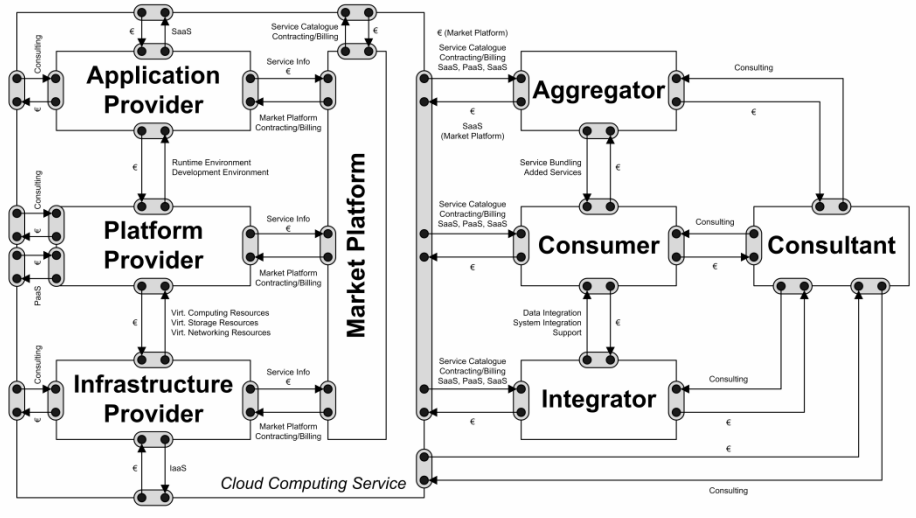


Fig. 1. A generic Value Network of Cloud Computing

The composite actor *Cloud Computing Service* represents the service as perceived by the *Consumer* who does not necessarily care how it is implemented and what other services are utilized in order to provide the requested service. Therefore the composite actor is comprised of the roles *Infrastructure Provider*, *Platform Provider*, *Application Provider* and *Market Platform*. Roles within this composite actor may offer objects jointly with other roles, but they may also offer objects on their own.

Consumers, *Aggregators* and *Integrators* may request any kind of services (SaaS, PaaS, IaaS) directly from one or more service providers or via a *Market Platform.* In Figure 1 this is represented by the value ports at the edge of the composite actor. For higher level services *Application* and *Platform Providers* can request services from other service providers. This is represented by the links among the service providers.

Both *Aggregators* and *Integrators* receive any number and kind of services (SaaS, PaaS, IaaS) from the composite actor to offer their value-added service to the *Consumer*. The *Consultant* is the only role that does not offer cloud computing services itself, but advises each of the other roles regarding cloud computing issues.

Empirical Findings on the Value Network

The interviews conducted throughout this research, all confirmed our identified roles and their relationships within the value network. The interviews also indicated two emerging roles, *Data Providers* and *Monitors*. A data provider would generate data and information to provide it for other actors within this network. A similar role called content provider is mentioned by Altmann et al. [3]. The monitoring role

provides permanent control of data privacy and security. Thereby, it controls the end-to-end connection, beginning with the first provider reaching to the consumer.

Exemplary Illustration
As described above and observed in practice, one company can act in more than one role. Salesforce.com is a typical example for a company that acts in different roles. On the one hand, and that is how they started, they offer a customer relationship management (CRM) solution[2] as SaaS, making them an *Application Provider*. With their Force.com[3] platform on the other hand they also offer a development and runtime environment to developers, taking on the role of a *Platform Provider*. Developers can either be *Application Providers* if they sell their applications to others, or *Consumers*, in case they just operate the application for own purposes. These third party cloud computing services developed and deployed on the Force.com platform can be offered on Salesforce.com's AppExchange[4] marketplace. Thus it also acts as *Market Platform*. From an external point of view Salesforce.com is no *Infrastructure Provider*, because it does not offer IaaS externally. To run its own and third-party applications on their platform it can either operate its own hardware or use the services of one or more *Infrastructure Providers*. To complete the example, *Aggregators* might take two or more cloud computing services, not all necessarily running on the Force.com platform, to build an aggregate solution that is useful for a certain group of *Consumers* or other service providers. *Integrators* on the other hand might introduce Salesforce.com's CRM solution into a production company, develop a customized solution on the Force.com platform on behalf of the *Consumer* and integrate all together with the on-premise SAP software. The *Consultant* could have prepared the way for the *Integrator* by, for example, advising the *Consumer* about the benefits of a cloud solution at an earlier stage of the introduction project.

6 Value Creation and Value Flow

Besides the validation of the conceptual generic value network, our empirical research also aimed at providing some first estimates on value creation and value flow within the network. Many interviewees compared the cloud computing market with the cell phone market. There are only a few big providers and many brokers, who buy network capacity and resell it under their label. Transferred to cloud computing, this could give a rise to the aggregator role, bundling existing solutions and reselling them with added value.

Two rivaling opinions arose. One group thought that especially the application provider and the integrator will generate most of the monetary value, since they need a deep understanding of the consumers' business model and their processes to offer solutions. Furthermore, they would profit from direct contact to the consumer, which can lead to follow-up projects. More than half of the interviewees argued that infrastructure and platform services will become a commodity. Thus they will only be profitable for high volume low margin businesses.

---

[2] http://www.salesforce.com/de/crm/
[3] http://www.salesforce.com/de/platform/
[4] http://sites.force.com/appexchange/

The second cluster of interviewees assigned the largest share of value creation to the infrastructure and platform providers. They argue that in many sectors such as the financial sector, applications are not very complex, but need large hardware resources.

7 Conclusion

As our discussion has shown, we believe that the value chain concept is too restricted to describe the cloud computing ecosystem with all its interrelationships between different market actors. Thus we postulate the application of a value network instead. We have described eight generic roles currently being observable in the cloud computing market and developed a generic value network to further analyze the emergent ecosystem.

Although, we could gain some first insights on value creation and value flow within the cloud computing ecosystem from our exploratory interviews, no valid estimations can be made yet. Therefore future research needs to investigate this in more depth on a broader empirical basis. Our proposed generic cloud computing value chain can serve as an empirically validated conceptual analytical framework to guide this research. Future research could also investigate the emerging data provider and monitor roles, reflecting them in practice to extend the generic value network.

From a practitioner's point of view, our proposed value network can be applied to strategically position a company or service offering in the cloud computing market and to identify possible business opportunities. Therefore it is not necessarily important to know, which generic role might take the largest share within the ecosystem, but to develop a unique value proposition based on core competencies.

Acknowledgements. The authors gratefully acknowledge the financial support for this research from Siemens IT Solutions & Services in the context of the Center for Knowledge Interchange at Technische Universität München. This research is part of the SIS-TUM competence center "IT Value Innovations for Industry Challenges".

References

1. Freiberger, P., Swaine, M.: Fire in the valley: the making of the personal computer. McGraw-Hill, New York (2000)
2. Stanoevska-Slabeva, K., Talamanca, C., Thanos, G., Zsigri, C.: Development of a generic value chain for the Grid industry. In: Veit, D.J., Altmann, J. (eds.) GECON 2007. LNCS, vol. 4685, pp. 44–57. Springer, Heidelberg (2007)
3. Altmann, J., Ion, M., Bany Mohammed, A.: Taxonomy of grid business models. In: Veit, D.J., Altmann, J. (eds.) GECON 2007. LNCS, vol. 4685, pp. 29–43. Springer, Heidelberg (2007)
4. Lee, H., Leem, C.: A study on value chain in a ubiquitous computing environment. In: Gervasi, O., Gavrilova, M.L., Kumar, V., Laganá, A., Lee, H.P., Mun, Y., Taniar, D., Tan, C.J.K. (eds.) ICCSA 2005. LNCS, vol. 3483, pp. 113–121. Springer, Heidelberg (2005)

5. Mohammed, A., Altmann, J., Hwang, J.: Cloud Computing Value Chains: Understanding Businesses and Value Creation in the Cloud. In: Brazier, F.M.T., Rana, O.F., Strassner, J.C. (eds.) Economic Models and Algorithms for Distributed Systems, pp. 187–208. Birkhäuser, Basel (2010)
6. Porter, M.E.: Competitive advantage: creating and sustaining superior performance. Free Press, New York (1985)
7. Youseff, L., Butrico, M., Da Silva, D.: Toward a Unified Ontology of Cloud Computing. In: Grid Computing Environments Workshop, pp. 1–10 (2008)
8. Leimeister, S., Riedl, C., Böhm, M., Krcmar, H.: The Business Perspective of Cloud Computing: Actors, Roles, and Value Networks. In: 18th European Conference on Information Systems, ECIS 2010, Pretoria, South Africa (2010)
9. Sturgeon, T.J.: How do we define value chains and production networks? MIT IPC Globalization Working Paper 00-010 (2001)
10. Peppard, J., Rylander, A.: From Value Chain to Value Network: Insights for Mobile Operators. European Management Journal 24, 128–141 (2006)
11. Conolly, S.D., Matarazzo, J.M.: Knowledge and Special Libraries. Butterworth-Heinemann, Woburn (1999)
12. Bovet, D., Martha, J.: Value Nets: Breaking the Supply Chain to Unlock Hidden Profits. John Wiley and Sons, New York (2000)
13. Normann, R., Ramirez, R.: From Value Chain to Value Constellation: Designing Interactive Strategy. Harvard Business Review 71, 65–77 (1993)
14. Stabell, C.B., Fjeldstad, O.D.: Configuring Value for Competitive Advantage: On Chains, Shops, and Networks. Strategic Management Journal 19, 413–437 (1998)
15. Biem, A., Caswell, N.: A value network model for strategic analysis. In: 41st Hawaii International Conference on System Sciences, HICSS (2008)
16. Gulati, R., Nohria, N., Zaheer, A.: Strategic Networks. Strategic Management Journal 21, 203–215 (2000)
17. Pil, F.K., Holweg, M.: Evolving From Value Chain to Value Grid. MIT Sloan management review 47, 71–80 (2006)
18. Miller, J.G.: Living systems. McGraw-Hill, New York (1978)
19. Miles, R., Snow, C.: Organizations: New concepts for new forms. California Management Review 28, 62–73 (1986)
20. Miles, R., Snow, C.: Causes of failures in network organizations. California Management Review 34, 53–72 (1992)
21. Tapscott, D., Lowy, A., Ticoll, D.: Digital capital: harnessing the power of business webs. Harvard Business Press (2000)
22. Steiner, F.: Formation and early growth of business webs: modular product systems in network markets. Physica-Verlag, Heidelberg (2005)
23. Nambisan, S., Sawhney, M.: The global brain: your roadmap for innovating faster and smarter in a networked world. Wharton School Publishing (2007)
24. Barros, A., Dumas, M.: The rise of web service ecosystems. IT Professional 8, 31–37 (2006)
25. Muylle, S., Basu, A.: Online support for business processes by electronic intermediaries. Decision Support Systems 45, 845–857 (2008)
26. Currie, W.: The supply-side of IT outsourcing: the trend towards mergers, acquisitions and joint ventures. International Journal of Physical Distribution and Logistics Management 30, 238–254 (2000)
27. Haupt, S.: Digitale Wertschöpfungsnetzwerke und kooperative Strategien in der deutschen Lackindustrie. Doctoral Thesis, Universität St. Gallen, St. Gallen (2003)

28. Basole, R., Rouse, W.: Complexity of service value networks: Conceptualization and empirical investigation. IBM systems journal 47, 53 (2008)
29. Gordijn, J., Akkermans, H.: E3-value: Design and Evaluation of e-Business Models. IEEE Intelligent Systems 16, 11–17 (2001)
30. Riedl, C., Böhmann, T., Leimeister, J.M., Krcmar, H.: A Framework for Analysing Service Ecosystem Capabilities to Innovate. In: Proceedings of 17th European Conference on Information Systems, ECIS 2009, Verona, Italy (2009)
31. Riedl, C., Böhmann, T., Rosemann, M., Krcmar, H.: Quality Aspects in Service Ecosystems: Areas for Exploitation and Exploration. In: Proceedings of International Conference on Electronic Commerce (ICEC 2008), pp. 1–7. ACM, New York (2008)
32. Leavitt, N.: Is cloud computing really ready for prime time? Computer 42, 15–20 (2009)
33. Reeves, D., Blum, D., Watson, R., Creese, G., Blakley, B., Haddad, C., Howard, C., Manes, A.T., Passmore, D., Lewis, J.: Cloud Computing: Transforming IT In-Depth Research Overview, Burton Group (2009)
34. Guan, Y., Bao, J.: A CP Intrusion Detection Strategy on Cloud Computing. In: International Symposium on Web Information Systems and Applications (WISA), Nanchang, P. R. China, pp. 84–87 (2009)
35. Gentzsch, W.: Porting Applications to Grids and Clouds. International Journal of Grid and High Performance Computing 1, 55–77 (2009)
36. Briscoe, G., Marinos, A.: Digital Ecosystems in the Clouds: Towards Community Cloud Computing. Arxiv preprint arXiv:0903.0694 (2009)
37. Lin, G., Fu, D., Zhu, J., Dasmalchi, G.: Cloud computing: IT as a Service. IT Professional 11, 10–13 (2009)
38. Plummer, D.C., F., K.L.: Three Types of Cloud Brokerage will enhance Cloud Services. Gartner Research Report G00164265 (2009)
39. Böhm, M., Leimeister, S., Riedl, C., Krcmar, H.: Cloud Computing: Outsourcing 2.0 oder ein neues Geschäftsmodell zur Bereitstellung von IT-Ressourcen? Information Management & Consulting 24, 6–14 (2009)

SLA as a Complementary Currency in Peer-2-Peer Markets

Ioan Petri[1], Omer Rana[2], and Gheorghe Cosmin Silaghi[1]

[1] Business Information Systems, Babes-Bolyai University, Romania
[2] School of Computer Science & Informatics,
Cardiff University, UK

Abstract. Service Level Agreements (SLAs) provide a basis for establishing contractually binding interactions within Peer-2-Peer systems. Such SLAs are particularly useful when considering interactions in environments with limited trust between participants. Complementary currencies, on the other hand, have proven to be useful for facilitating exchange among selfish peers. We identify how an SLA can itself be used as a complementary currency to encourage resource sharing between peers. Our work demonstrates how an SLA can be used as a medium of exchange and used to establish a market for computational resources. The value of an SLA can vary based on demand for particular types of resources. Simulate a process of trade we investigate several economic indicators qualified for their significance in small economies. We demonstrate how the economic benefit (in terms of profit and loss) evolves based on varying levels of demand.

1 Introduction

Complementary currency systems have demonstrated to be highly effective in supporting exchange within a local context. For instance, in the context of a local economy (i.e. where trading is restricted to participants who are *nearby* – and does not involve global entities such as central banks), goods or services can themselves represent tradable objects. Complementary currency systems lead to autonomous economies where participants are able to generate their own currency. Successful experiments such as Comox Valley and Ithaca Hours [4] have proved the efficiency of such local currencies in real life contexts. Complementary currencies have also found applicability in Peer-2-Peer (P2P) networks as powerful instruments to promote exchange – particularly to avoid the need for accessing central servers. For instance, to make use of underutilised computing resources, mediums of exchange were introduced as local currencies allowing users to trade services with the confidence that the money earned can be exchanged for other services/resources. The concept of a complementary currency has been used to build unconventional local environments such as iWAT [8], Samsara [1], Geek Credit [6] or PPay [16]. Despite several technical difficulties, these studies demonstrate that complementary currencies provide an effective alternative charging mechanism within P2P networks.

J. Altmann and O.F. Rana (Eds.): GECON 2010, LNCS 6296, pp. 141–152, 2010.

A Service Level Agreement (SLA) provides a contract between a service provider and one or more users. An SLA contains guarantee terms that need to be satisfied by a provider, and a payment that needs to be made by a user when such guarantees have been met. As an SLA refers to a service that is to be provided in the future, it is possible to consider an SLA-based options market, and thereby use an SLA itself as a complementary currency. Whereas iWAT, by a ticketing mechanism, or Samsara by the mechanism of claims, require strategies to control the environment, SLAs provide such functionality by their design. An SLA therefore provides all the relevant features to serve as a model for an exchange process. In P2P systems using SLAs we can use economic benefit as a metric to identify whether profit or loss has been incurred based on an initial configuration at start time. The rest of the paper is organised as follows. In section 2 we analyse related approaches focusing on alternative currency systems. Section 3 presents an overview of an SLA as a complementary currency. A simulation using PeerSim is used in section 4 to validate our approach. Section 5 presents the results and summarises the findings of our research compared to the iWAT system.

2 Related Work

Complementary currencies provide an important mechanism to support the use of computational and data services. The WAT system [15], for instance, is a debt-oriented (i.e. a capability is used before payment) approach based on WAT tickets, and provides an alternative currency to facilitate local exchanges. A ticket may be issued by any participant and used for mediating exchanges. WAT has been extended in [12], [10], [9] to support strategies, security management, and fault tolerance respectively. iWAT [8] (the internet version of WAT) implements the WAT core and examines the problem of ticket exchange by designating the drawer as the responsible party for ticket authenticity. The system uses incentives to keep ticket drawers online and has no fixed authority for overseeing the exchange. It also implements an internal policy of trusted keys among users which ensures safe communication in the system. iWAT identifies two different approaches to vary the value of a ticket: (1) reduction and (2) multiplication. Reduction implies that the value of a ticket decreases over time, based on demand (multiplication is the opposite). A reduction in the value of tickets [13] accelerates spending, representing a sort of mutual help among users. It is demonstrated that ticket reduction accelerates trades and reduces bankruptcies while multiplication provides limited benefit for the drawers and must be used carefully. Saito et al. [13] illustrate the users tendency to deny losses and to defer redemption. They also accurately specify the expectations of the participants and the risks they incur. Instead of tickets, we instead use an SLA whose value can also change over time (using reduction and multiplication).

Samsara [1] concentrates on establishing service relationships between peers, minimising risk due to failure and claim, to ensure equivalence between the contribution and consumption of peer nodes. Establishing symmetric storage

relationships leads to fairness and is handled as a central goal. Samsara is similar to the WAT system, however although designed as a mechanism to support exchanging objects for storage replication, it is preoccupied with fairness rather than offer an accurate payment representation.

Token-based systems share many similarities with our work – where each peer holds an amount of tokens proportional to the value of the goods being purchased. The client must transfer a certain amount of tokens to the supplier based on the value of the service. Tokens can be created by a central organization and exchanged for real currency outside the local system (e.g. euros, dollars, etc.) For token-based systems, the value of a token must be consistent throughout the system. Each token is the minimal unit of payment and designed to manage the resource consumption. PPay [16] is a token-based system which implements a protocol for coin assignment at the level of peer nodes using signature generation, verification and network messages. In PPay each peer can buy coins from a bank, and then becomes the owner of the coin. To pay, one peer has to transfer the coin and therefore the ownership of that coin, to a new holder. In case the holder chooses to re-sell the coin, an approval from the owner is needed for changing again the ownership of the coin. PPay aims to support a secure and efficient payment process by implementing a strategy for coin transfer. By introducing an additional authority to approve exchanges and transform real money into specific PPay digital coins, PPay hypothesises that security should be a major concern in the context of digital currency systems.

The Ripple protocol [3] enables trustworthy "IOUs" to be exchanged between directly connected participants, defining particular trusted connections for each currency. Each IOU operates on a separate network and participants have the authority to accede in different currency networks. In Ripple it is assumed that a single resource sharing context has significance for all the participants. In most cases this assertion represents only an assumption; e.g. a particular resource has significance (utility) for A but probably not for B.

For alternative currency systems, trust implications have been considered in much more detail than the representation of the exchanged object [7], [9]. Another related issue is counteraction against *whitewasher* peers that may damage the exchange [11]. Whitewashers are defined as an undesirable category of participants that strategically leave and rejoin the system by changing identities and impact system welfare. MCS (Mutual Credit System) [14] is a debt-oriented model that allows participants to credit themselves. In the MCS and WAT systems, welfare is strongly related to the bankruptcy rates and the percentage of whitewashers. Five different evasive strategies are introduced in the context of iWAT ticket variation to reduce the cost of the exchange.

3 Motivation and Approach

An SLA may be used to specify quality of service terms, the measurement criteria, reporting criteria and penalty/reward clauses between participants. Within an electronic market, an SLA may be used for: (i) an expression/proof of debts as

well as credits – debts to the client and credits to the service provider; (ii) as a token of exchange between participants; (iii) as an identification of responsibilities of participants involved (such as the client and service provider). Establishing an SLA between two parties (client & service provider) implies that the service provider has agreed to provide a particular capability to the client within some quality of service. In return, the client must provide a monetary payment (most often) or credit to the provider once the service has been delivered (subject to a penalty, often also monetary, in case the quality of service terms have not been adhered to). The credit may be made at the beginning of service provision (the *pay-before-use* model) or after (the *pay-after-use* model). It is useful to note that an SLA refers to a service provision that must take place some time in the future. The SLA creation process starts when a client (the initiator) sends a request to a potential provider. The provider issues an SLA template, specifying agreement terms and obligations – containing service level objectives, quality terms and business values associated with particular service level objectives. Penalties and rewards are also parts of the SLA template. The initiator fills the template with the required service, asking the provider for a price. The agreement is finalised when the initiator accepts the price from the provider. The underlying protocol can be found in the WS-Agreement specification [2].

3.1 Protocol

We assume that a provider can deliver two categories of services within a trusted environment: (i) those that it provisions through it's locally owned resources (direct provisioning); (ii) those that it defers to other providers through previously acquired SLAs (indirect provisioning). The first category corresponds to SLA creation while the second involves advertising SLAs that a provider already holds with others.

An SLA life cycle consists of the following stages (see figure 1): (1) SLA issuing $t_0 - t_1$, (2) SLA forwarding $t_1 - t_2$, $t_2 - t_3$ and SLA redemption t_3. Issuing reproduces the economic process of supplying the system with exchangeable units (SLAs), whereas redemption represents the final stage of acquiring the service identified in the SLA. Between these there can be multiple stages involving SLA forwarding between nodes.

Issuing: from figure 1 node A, the initiator, asks node B for the services it can provide (directly or indirectly). In figure 1, $[T(B), \{t_1, t_2, ..., t_n\}]$ identifies the list of services that node B can provide, where $t_i = SLA_B^X$ is an SLA that node B has with node X, and which node B is willing to trade. Hence, node B advertises both services it can offer, along with those SLAs that it holds with other nodes. While node A is able to chose the properties (quality terms) for capability that node B provides, it cannot modify the properties of an existing SLA that node B holds (which is advertised on a *take-it-or-leave-it* basis). Node A chooses to acquire the service that node B can provide directly, therefore B forwards the agreement template T(B). After creation, node B commits to provide the service in T(B) and node A credits node B with the monetary value identified in the SLA before service provisioning begins. At this point node A

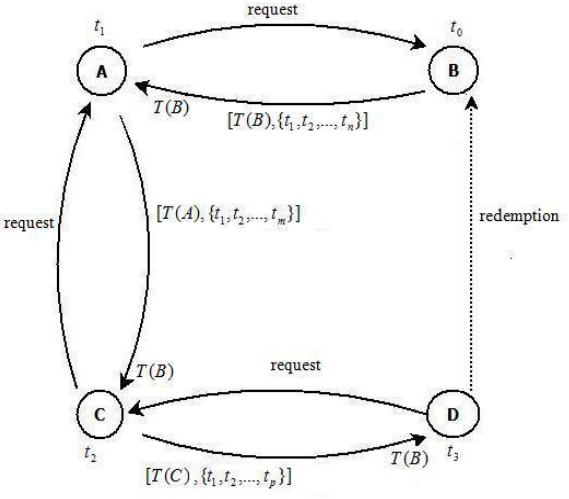


Fig. 1. Protocol Workflow

holds an SLA with B (SLA_A^B) which has a monetary value (equivalent to what node A has paid node B). Node A can now present the service specified by SLA_A^B in its own list of available services that can be provided indirectly. The value of this SLA, however, will change over time depending on the demand for services being provided by B. The SLA is therefore a token of exchange that may be used by A to purchase additional capability from other nodes – if node A does not use this capability directly.

Forwarding: in the second stage of the exchange node C now requests a service from node A. Node A responds with the list $[T(A), \{t_1, t_2, ..., t_m\}]$. Within this list T(A) is the template specifying the service that node A can provide directly. Node C chooses to acquire the service specified in t_1 and pays A based on the advertised price for this SLA. Node B must now provide a service to C in accordance with the SLA that was previously agreed between nodes A and B. The client of the agreement has changed, as SLA_A^B becomes SLA_C^B, and the value of SLA_A^B is different from SLA_C^B. After node A forwarded the SLA to C, a transfer of obligations from A to C is invoked. Therefore, the provider (node B) has an obligation for the current holder (node C) of the SLA; as we assume a trusted environment, the redeemer of an SLA can change to any trusted participating nodes. At this point node C is able to list the service defined by this SLA as a service that can be provided indirectly.

Redemption: this is the final stage of the exchange and involves using the capability identified in the SLA (it occurs after an SLA has been agreed between two parties, and may take place after several forwarding operations). As illustrated in figure 1 node D redeems the SLA with provider B.

3.2 Costs

The exchange process runs with a specific distribution of value among the participating nodes. Therefore, after the issuing stage $(t_0 - t_1)$ node A holds an SLA valued at $P_{SLA_A^B}$=10mu (monetary units) with node B. Whereas node B has already been credited, node A has a token worth 10mu it can now exchange with other nodes. As the value of an SLA can change over time, in the forwarding stage, the SLA can either decay or increase; this is based on the current demand for services of node B in the market.

4 Simulation Design

To validate our use of an SLA as a complementary currency, we use a PeerSim [5] based simulation. PeerSim is an open source, Java-based simulation framework for developing and testing P2P algorithms in a dynamic environment. A PeerSim-based simulation allows us to develop hypotheses that can be applied in the context of complementary currencies. The simulation is developed within a particular network infrastructure where peers can play different roles: issuers, recipients or redeemers. The value of an SLA can change over time, and we use a Poisson distribution to model this increase (or decay). In order to express a number of exchange events that can occur in a fixed period of time, a Poisson distribution is used. The Poisson distribution is applied when counting the number of discrete events and the random variable can only take non-negative integer values. It is closely connected to the exponential distribution, which (among other applications) is used to measure the time between arrivals of the events. The Poisson distribution was chosen from the need to simulate expected events (SLA processes), that occur independently, have a low frequency in the system and the number of participants is high.

Every SLAs has: (i) An issuer (I_s); (ii) a specific price (P_s); (iii) a creation time T_s; (iv) an (age); and (v) A time to live (ttl). As an SLA has an expiry time (age) associated with it, there is a limited time over which an SLA can be exchanged with other nodes. During this time interval the SLA can either increase or decrease in value, based on the prevailing market conditions.

Each peer node has a unique identifier in the simulation. They can have multiple states and are designed to be repositories of services – although only one service can be provisioned at a specific time. The Poisson distribution specifies the service that a node can provide during the experiment. It is assumed that a node acquires only those SLAs that represent an immediate benefit. An immediate benefit is the SLA that brings a certain level of welfare or contains a desired level of service. During the exchange one node can perform the following operations: submitting or issuing SLAs at each cycle according to a predefined configuration, spending or forwarding SLAs and redeeming SLAs.

4.1 Configuration

The algorithm is cycle-based, one node being scheduled to process one specific operation (issue, forward or redemption) at each cycle. During the experiment

the protocol uses the following distribution: (i) Simulation *cycles* $c_i, i = \overline{1,n}$: number of cycles over which the experiment executes. At cycle $c_k, k < n$, the system records a level of benefit expressed in terms of profit P_k and loss L_k; (ii) *size*: the number of nodes in the experiment; (iii) *time to live(ttl)*: the maximum allowed age for an SLA; (iv) the network *topology*: a specific network architecture. By default the system uses a random connection topology; (v) *nodes* define the number of peers issuing SLAs; (vi) *interval*: average time interval between SLAs submissions; and (vii) *number*: maximum number of SLAs the system uses during the experiment.

4.2 Protocol Execution

As SLAs are submitted according to a predefined probability distribution, the experiment requires sufficient cycles to ensure an adequate number of submissions have been made. The algorithm handles redemption as an operation scheduled to interfere at a specified *ttl* time. The operation of the algorithm in terms of issuing, forwarding and redeeming is identified in section 1. These operations are carried out over a duration specified in the time to live (*ttl*) parameter. When SLA's *age* exceeds the specified *ttl* parameter the protocol ends by a redemption.

Algorithm 1. Protocol Execution

```
1: for all i nodes from network size do
2:    while ttl < age and i < submissionnodes do
3:       issue (node, SLA);
4:    end while
5: end for
6: for all SLAs do
7:    if ttl < age then
8:       forward(node, SLA);
9:    else
10:      redeem(node, SLA);
11:   end if
12: end for
```

It is assumed that redemption is automatically invoked when $age = ttl$. The experiments prove that deferring redemption represents a factor of immediate benefit for peers.

The economic benefit in terms of profit and loss: The benefit is defined as a function $f_x : X \rightarrow B$ where X is a set containing values specified by a configuration, and B is a set of peers that have made a profit or loss after multiple exchanges. Profit is defined as an auxiliary function defined by $f_x : X \rightarrow P_s$, where X is a set of configuration parameters and P_s is a list of peers in profit. Similarly, loss is defined as $f_x : X \rightarrow L_s$, where X is a configuration and L_s is a list of peers who have made a loss.

During the exchange two operations can occur at a node: (a) buying (acquiring) SLAs with costs: (C_s) as a sum of SLA object prices during stages of exchange: $C_s = \sum_{i=1}^{k} P_{SLA_i}$; where P_{SLA} represents the price of one acquired

Algorithm 2. Tracking Node Benefits

```
1: for all cycles c do
2:   for all i nodes in scope do
3:     calculate I_s and C_s
4:     if I_s ≥ C_s then
5:       profittoken = true;
6:       return profit ++;
7:     else
8:       losstoken = true;
9:       return loss ++;
10:    end if
11:  end for
12: end for
```

SLA; k represents the number of SLAs that are acquired during the experiment; (b) selling (spending) SLAs with income: (I_s) as a sum of SLA prices that one node sells during stages of exchange: $I_s = \sum_{i=1}^{k} P_{SLA_i}$; where P_{SLA} represents the price of a sold SLA; k represents the number of SLAs that are sold during the experiment. According to a probability distribution the system can express statuses by: Profit: $P_s = I_s - C_s$; and Loss:$L_s = C_s - I_s$. The algorithm handles the output by counting the number of peers in profit or loss after each cycle. In the simulation, an *observer* collects the information from various experiment stages and to provide the simulation output.

5 Results

Our approach uses an unstructured P2P architecture with peers operating services in the network. Each peer can deliver several services and each service has a number of associated parameters. Services are delivered on the network on the basis of established SLAs among peers. We consider that every node holds a service distribution and it is linked with other nodes building virtual organisations. One peer has a (i) predefined structure of links to neighbours, (ii) a budget limit. Our algorithm uses a *pay-before-use* payment scheme. It means that every service delivery implies an advance payment. The simulations use a limited number of participants $p_k \in P$, where P is a set containing nodes from the network. The model has a random network topology where participants are linked together by some predefined paths. Each participant can reach others after some numbers of hops. Within the system, one participant can build an initial view of potential partners according to SLA properties (e.g. one node builds a view of all peers able to provide a particular type of Operating System or processor).

5.1 Scenarios

Various experiments (scenarios) are conducted to validate the system. In Experiment 1 we demonstrate how economic benefit is influenced by the submission interval of service requests – the submission interval is incremented from 1

to 10. The experiment uses a configuration with 10 issuing nodes and a maximum of 10 SLAs per node. Figure 2 confirms that benefit is higher when SLAs are submitted with a time interval of 1 (at each cycle) – there are always more SLAs in the system (over the monitoring interval), therefore the level of benefit is higher. With the submission interval of 10, the SLAs are injected with some latency (randomly at each 10 cycles) and the system reaches the maximum level after several cycles of exchange. We note that benefit increases continuously after 60 cycles, indicating that benefit is higher when SLAs are submitted over shorter time intervals, leading to greater overall circulation of SLAs within the system. For a real economy, the experiment validates that a high frequency distribution leads to growth in terms of benefit. The highest level of benefit occurs when $\varphi = 1$, where $\varphi = \overline{1, 10}$ is the distribution frequency.

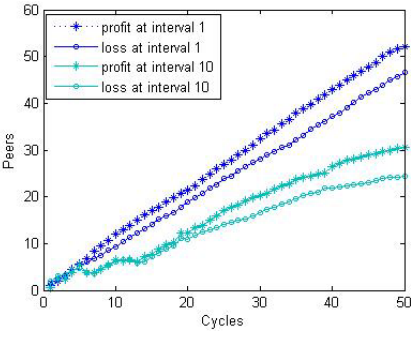


Fig. 2. Submitting SLAs at different intervals

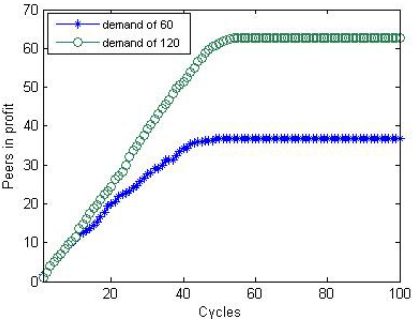


Fig. 3. Profit generated at different levels of demand

Experiment 2: the economic benefit when demand is increased. This experiment (figure 3) investigates how the economic benefit, expressed as the total number of profitable peers, evolves when the number of requesting participants is increased. Figure 3 show the profit and loss trend in the first 10 cycles. For the first 10 cycles the system has enough requesting peers to operate trades. After cycle 10 the requests cause a gradual growth from cycle 10 until cycle 50. The growth trajectory then becomes linear as new participants are able to satisfy demand.
Experiment 3: he economic benefit at different types of SLAs covering demand. These experiments reflect how the economic benefit is influenced by supporting particular types of SLAs. According to the SLA type, the system can record either profit or loss. These scenarios try to validate the relation between the benefit and SLA types used to cover service demand. The following configurations use 10 issuing nodes and the number of SLAs is varied.

Figure 4 presents 4 different situations when demand is matched with an increasing number of SLAs. For 5 SLAs the system reveals a certain level of benefit. Further the output varies according to the number of participants. Because the number of decaying SLAs is stable, the overall level of loss remains the same. In

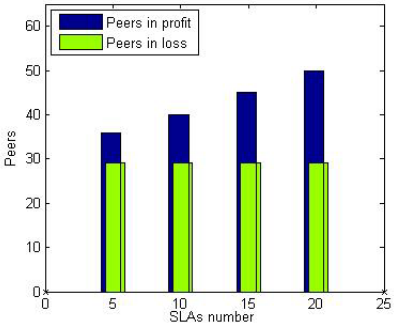


Fig. 4. Matching demand with SLA generation

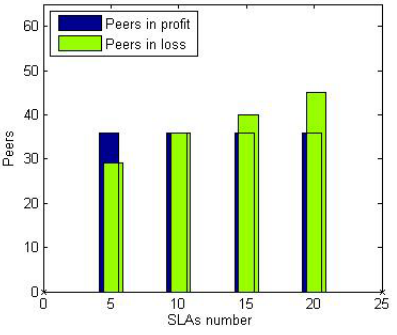


Fig. 5. Economic benefit with decaying SLA value

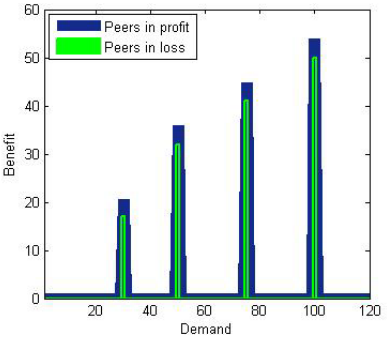


Fig. 6. Simulating economic benefit at different levels of demand

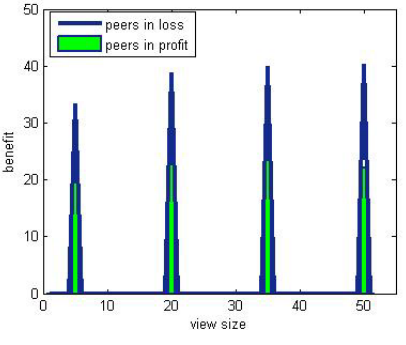


Fig. 7. Simulating economic benefit for different peer views

the following experiment (figure 5) the number of decaying SLAs used to cover demand is increased in order to validate the connection between decaying SLAs and the system's benefit. With these experiments we validated the assumption that the economy status is closely related to SLA types.

Experiment 4: the economic benefit at different stages of demand. This scenario (figure 6) records overall system behaviour when the number of participating nodes change. Particularly, we handled demand as the number of peers that request services. The experiment uses 10 nodes generating SLAs and a maximum of 10 SLAs per node. Figure 6 displays a proportional representation between the economic benefit and the level of demand measured in number of peers requesting SLAs. The experiment reveals the dependency between profit, loss and demand. It is observed that new levels of demand cause new levels of benefit. Therefore, keeping a fixed number of participants, the system records values of 20 to 55 peers in benefit for levels of demand defined over the interval [25,100].

Experiment 5: the economic benefit at different views of peers, where one *view* represents a collection of peers selected as potential trading partners. By modifying the number of proximate traders, we are able to induce a level of benefit. We are also interested in identifying how this *view* impacts the overall system benefit. The experiment uses a configuration with 10 nodes generating SLAs and a maximum of 10 SLAs per node.

The experiment shows that expanding the overall view (i.e. possible potential trading partners) for one trader causes benefit. As a trader is able to interact with additional partners, the overall exchange within the system improves – leading to greater benefit. Figure 7 illustrates benefit when the view oscillates from 5 to 20 peers. Although there is a maximum number of peers after which benefit does not increase further. In this case, we are able to observer that for more than 20 peers in view the benefit remains stable.

6 Conclusions

We use an SLA as a token of exchange between participating nodes. As an SLA represents provisioning that must take place in the future, an SLA may be circulated within a community of trusted peers. This may occur either because a peer speculates that the capability offered by another peer may be at a higher demand at some time point in the future, or if a peer no longer needs the capability that it requires from another peer (but for which an SLA has already been generated). Trusted, in this context, implies that any client within the domain can redeem the SLA, and not necessarily the initiating peer. SLAs may also be used in this way by brokers for aggregating capacity from various peers within a network.

Using simulation, we demonstrate that system benefit is influenced by the number of participants, the level of demand and the type of SLAs used. The experiments show that the view of peers (i.e. how many partners a peer can trade with), as a configuration parameter, induces benefit. Further, it was demonstrated that the system experiences a level of benefit directly related to the SLA submission interval – indicating that a higher circulation (i.e. lower submission interval) in the system improves overall benefit.

Acknowledgements

We acknowledge support of the Romanian National Authority for Scientific Research under the project $IDEI_5$73. Ioan Petri acknowledges support by the Sectorial Operational Programme Human Resources Development, Contract POSDRU 6/1.5/S/3: "Doctoral studies: through science towards society".

References

1. Cox, L.P., Noble, B.D.: Samsara: Honor Among Thieves in Peer-to-Peer Storage. In: In Proc. of 19th ACM Symposium on Operating Systems Principles, pp. 120–132. ACM Press, New York (2003)

2. Open Grid Forum: WS-Agreement Specification (March 14, 2007), http://www.ogf.org/documents/GFD.107.pdf
3. Fugger, R.: Money as IOUs in Social Trust Networks and a proposal for a Decentralized Currency Network protocol (2004), http://ripple.sourceforge.net/
4. Glover, P.: Ithaca Hours, http://www.ithacahours.com
5. Jelasity, M., Montresor, A., Jesi, G.P., Voulgaris, S.: The Peersim simulator, http://peersim.sf.net
6. Komarov, A.: Geek Credit, http://home.gna.org/geekcredit/
7. Moreton, T., Twigg, A.: Trading in Trust, Tokens, and Stamps. In. In: Proc. of the First Workshop on Economics of Peer-to-Peer Systems (2003)
8. Saito, K.: i-WAT: The Internet WAT System – An Architecture for Maintaining Trust and Facilitating Peer-to-Peer barter Relations. PhD thesis, School of Media and Governance, Keio University (February 2006)
9. Saito, K.: WOT for WAT: Spinning the Web of Trust for Peer-to-Peer Barter Relationships. IEICE Trans. on Communication 88, 1503–1510 (2005)
10. Saito, K., Morino, E.: Local production, local consumption storage economics for peer-to-peer systems. In: Proceedings of the 2008 International Symposium on Applications and the Internet. IEEE Computer Society Press, Los Alamitos (2008)
11. Saito, K., Morino, E.: The brighter side of risks in peer-to-peer barter relationships. Future Generation Computer Systems. Corrected Proof (in Press, 2010)
12. Saito, K., Morino, E., Murai, J.: Incentive-Compatibility in a Distributed Autonomous Currency System. In: Despotovic, Z., Joseph, S., Sartori, C. (eds.) AP2PC 2005. LNCS (LNAI), vol. 4118, pp. 44–57. Springer, Heidelberg (2006)
13. Saito, K., Morino, E., Murai, J.: Reduction Over Time: Easing the Burden of Peer-to-Peer Barter Relationships to Facilitate Mutual Help. In: Computer Supported Activity Coordination, pp. 28–37. INSTICC Press (2005)
14. Schraven, J.: Mutual Credit Systems and the Commons Problem: Why community currency systems such as LETS need not collapse under opportunistic behavior. Int. Jour. of Community Currency Research 5 (2001), http://www.geog.le.ac.uk/ijccr/
15. watsystems.net. Watsystems homepage, http://www.watsystems.net
16. Yang, B., Garcia-Molina, H.: PPay: Micropayments for Peer-to-Peer Systems, pp. 300–310. ACM Press, New York (2003)

SLA Validation in Layered Cloud Infrastructures

Irfan Ul Haq[1], Ivona Brandic[2], and Erich Schikuta[1]

[1] Department of Knowledge and Business Engineering,
University of Vienna, Austria
{irfan.ul.haq,erich.schikuta}@univie.ac.at
[2] Distributed Systems Group, Information Systems Institute
Vienna University of Technology, Vienna, Austria
ivona@infosys.tuwien.ac.at

Abstract. Cloud computing brings in a novel paradigm to foster IT-based service economy in scalable computing infrastructures by allowing guaranteed on-demand resource allocation with flexible pricing models. Scalable computing infrastructures not only require autonomous management abilities but also the compliance to users' requirements through Service Level Agreements (SLAs). Such infrastructures should *automatically respond* to changing components, workload, and environmental conditions as well as *prevent violations* of agreed SLAs. The essential requirements for SLA-based orchestration of services include agile component-based infrastructure to support these orchestrations; proactive validation of SLAs to prevent violations; and business enabling requirements including trust, privacy and breach management to address penalty enforcement, renegotiation and recovery. In this paper we outline a systematic approach that weaves together three different SLA management models, which cater the above mentioned issues by localizing and addressing them at different but interrelated scopes pertaining to resource, infrastructure, and business domains. In our framework Cloud components (e.g., meta negotiator, broker, automatic service deployer) are loosely coupled using SLAs and can be exchanged on demand considering current load, systems failures, and the whole Cloud ecosystem. Thereafter, SLAs are validated based on high level goals (e.g., business rules, VO policies) ensuring trust, privacy, and breach management in layered Cloud infrastructures.

1 Introduction

Cloud computing can be defined in terms of the convergence and the evolution of several concepts from virtualization, distributed application design, Grid and enterprise IT management to enable a more flexible approach for deploying and scaling applications [1]. Service Level Agreements play an important role in connection with the service provisioning in Grid and Cloud based systems. An SLA facilitates automatically processable contract signed between the customer and the service provider that guarantees a specified Quality of Service (QoS) and enforces penalties in case of agreement violation.

In eBusiness platforms, SLA is essentially important for the service consumer as it compensates the consumer's high dependency on the service provider. With

J. Altmann and O.F. Rana (Eds.): GECON 2010, LNCS 6296, pp. 153–164, 2010.

the advent of Cloud computing, there is a high potential for third party solution providers such as Composite Service Providers (CSP), aggregators or resellers [1] to tie together services from different Clouds or external service providers to fulfill the pay-per-use demands of their customers. A cumulative contribution of such Composite Service Providers will emerge as service value chains. SLAs guarantee the expected quality of service (QoS) to different stakeholders at various levels in these service value chains. A major requirement in this regard thus is to foster SLA-based hierarchical service composition scenarios and their underpinning business networks and supply chains. This is a multi-facet challenge targeting SLA management both from technological and business points of view.

The major question at infrastructure-level is how to build a foundational infrastructure to allow SLA-based cross-enterprize service value chains and their validation, which is capable of doing infrastructure level adjustments to sustain the overall desired Quality of Service (QoS). Another requirement at resource-provision-level is a runtime monitoring and validation mechanism that maintains low level resource metrics in conformance with the high level Service Level Objectives. Challenges at business-level requires business concerns such as privacy, trust security and value sustainment to be kept intact during service aggregation and validation and to allow user-driven validation queries and penalty enforcement in case of under-performing services.

From infrastructure and resource viewpoints SLA management in Clouds [7] discusses strategies such as autonomic resource management [6], violation propagation in layered clouds [4] and the similar. However, there is still a lot to be done on facilitating SLA management in Clouds from the business perspective for example in terms of privacy, trust and security requirements to enable business value networks [10] [13].

In this paper we motivate from a scenario based on service value chains to realize the above mentioned issues in Cloud Computing's resource, infrastructure and business levels. Following the scenario we integrate our three validation models specially designed to target the resource, infrastructure and business requirements, and propose a holistic approach to SLA validation in Cloud based service value chains.

The holisitic SLA validation framework:

- complements an infrastructure that allows scalable, exchangeable and self adaptable components thus engineering at infrastructure-level, a basis for cross-enterprize service orchestrations in Clouds. This is based on the LAYSI infrastructure [4].
- offers autonomically manageable resource provision thus synchronizing low level resource provision with Service Level Objectives. This is taken care by the LoM2HiS SLA management model [5].
- ensures business essentials such as privacy, trust and SLA compliance thus catering business-level abstractions. It is based on our SLA Aggregation and Validation Framework [13].

In section 2 we present a motivational scenario to highlight SLA requirements from system's point of view. Section 3 discusses our three validation models at

resource, infrastructure and business levels respectively and section 4 integrates these models into a holistic validation framework. Section 5 concludes the paper with a a summary of work and future plans.

2 Motivational Scenario

Service value chains in Cloud based infrastructures represent an adequate example to understand the issues and requirements of an SLA validation framework from infrastructure, resource, and business point of views. In this section, we highlight various requirements of SLA validation for Layered Cloud Infrastructure with the help of a scenario, based on service value chains.

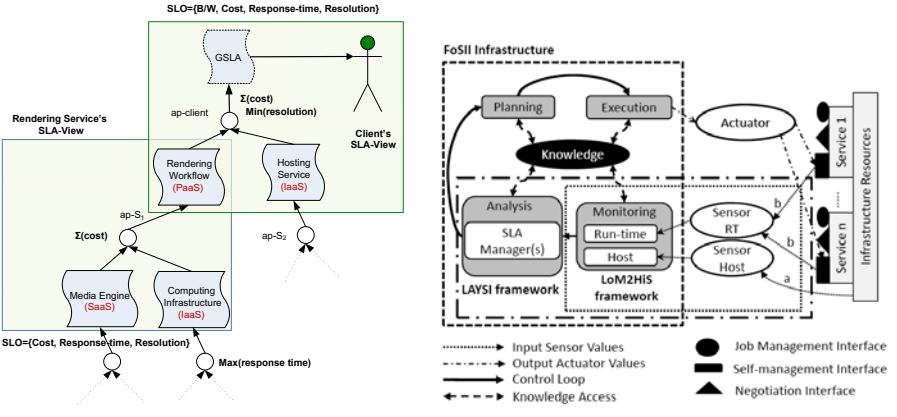


Fig. 1. (a) Motivational Scenario, (b) LoM2HiS Integrated within the FoSII Infrastructure

The motivational scenario depicted in figure 1(a) deals with dynamic workflow composition of Cloud based services. Arfa is a graphics designer and she has just finished designing an animation involving thousands of high resolution images. Now she needs to carry out hi-tech multi-media operations such as rendering and editing. She plans to utilize online services to accomplish these tasks. A media workflow service allows Arfa to define a series of activities involving video rendering, compression etc. Afterwards she would like to host the final compressed video on a dedicated server to visualize the output of her work. From Arfa's view-point she is only composing two workflow services, i.e. "rendering workflow" service and the "hosting service" characterized as a "Platform as a Service" (PaaS) and an "Infrastructure as a Service" (IaaS) respectively by her Cloud based service providers. What Arfa's View does not cover is the fact that both of these services themselves result from an aggregation of even further basic services, extending a supply chain type of structure beneath them. Similarly

the rendering workflow subdivides into services such as the "media engine" and the "computing infrastructure" provided by different service providers in a public Cloud. The QoS of services is guaranteed through SLAs and in case of any shortcomings, a penalty is promised.

To realize this scenario, we need:

- an enabling infrastructure that supports SLA-based service discovery, negotiation and orchestration.
- client-driven SLA validation for orchestration consistency as well as violation detection and penalty enforcement
- service provider-driven proactive SLA validation for violation prevention.
- runtime infrastructure level adjustments to sustain the guaranteed level of service.

In the following section we discuss three SLA validation models, which facilitate these requirements.

3 SLA Validation in Layered Cloud Infrastructures

In this section we discuss LoM2HiS [5] - a framework for user driven monitoring of SLAs in Cloud infrastructures, LAYSI [4] - a framework for SLA based layered Cloud infrastructures and the Rule-Based [13] Validation models for resource, infrastructure and business level SLA validation requirements.

3.1 Resource-Level SLA Validation: LoM2HiS

LoM2HiS (Low Level Metrics to High Level SLAs) system provides a means to map resource metrics to high level service parameters. The service provider in this way uses this system to maintain the contracted QoS. It is an integral component of the *Foundations of Self-governing ICT Infrastructures (FoSII)* project [6].

As shown in Figure 1(b), the *FoSII* infrastructure is used to manage the whole lifecycle of self-adaptable Cloud services [6]. Each FoSII service implements three interfaces: (i) negotiation interface necessary for the establishment of SLA agreements, (ii) job-management interface necessary to start the job, upload data, and similar job management actions, and (iii) the self-management interface necessary to devise actions in order to prevent SLA violations. The self-management interface shown in Figure 1(b) is implemented by each Cloud service and specifies operations for sensing changes of the desired state and for reacting to those changes. Therefore all the services shown in figure 1(a) implement LoM2HiS interface. As depicted in Figure 1(b), the host monitor sensors continuously monitor the infrastructure resource metrics and provide the autonomic manager with the current resource status. The run-time monitor sensors sense future SLA violation threats based on resource usage experiences and predefined threat thresholds. The threat thresholds are retrieved by the knowledge management system considering historical system experiences [3]. The mapping

between the the sensed host values and the values of the SLA parameters is described next.

As shown in Figure 1(b) we distinguish between the *host monitor* and the *runtime monitor*. Resources are monitored by *host monitor* using arbitrary monitoring tools (e.g. ganglia). Thus, resources metrics include e.g., down-time, up-time, available in and out bandwidth. Based on the predefined mappings stored in a database monitored metrics are periodically mapped to the SLA parameters. For example service availability Av of the storage service associated with the Computing Infrastructure shown in figure 1(a) is calculated using the resource metrics *downtime* and *uptime* during a given period of time (say 24 hours) and the according mapping rule looks like the following one:

$$Av = 1 - downtime/uptime \tag{1}$$

The mapping rules are defined by the provider using appropriate Domain Specific Languages (DSL). These rules are used to compose, aggregate, or convert the low-level metrics to form the high-level SLA parameter including mappings at different complexity levels e.g., $1 : n$ or $n : m$. Thus, calculated SLA values are compared with the predefined threat threshold in order to react before SLA violations happen. The concept of detecting future SLA violation threats is designed by defining a more restrictive threshold than the SLA violation threshold. Generation of the *threat threshold* is far from trivial and should be defined and mananaged by the FoSII's knowledge management system [3]. Once the possible SLA violations are detected, reactive action has to be taken in order to prevent SLA violations.

3.2 Infrastructure-Level SLA Validation: LAYSI

LAYSI - A Layered Approach for Prevention of SLA-Violations in Self-manageable Cloud Infrastructures, is embedded into the FoSII project (Foundations of Self-governing ICT Infrastructures) [6], that aims at developing self-adaptable Cloud services. LAYSI provides an agile component based architecture for layered cloud infrastructure and facilitates SLA-based service discovery, orchestration, maintenance and fault tolerance. The layered Cloud architecture utilizes loosely coupled and replaceable components like negotiator, broker or automatic service deployer which are hierarchically glued together through SLAs. The LAYSI framework also facilitates future SLA violation detection and propagation of reactive actions to the appropriate layer of the Cloud infrastructure. For decision making, we use knowledge databases proposing reactive actions by utilizing case based reasoning - a process of solving problems based on past experience. Based on the novel communication model we present how possible SLA violations can be identified and propagated to the layer of the Cloud infrastructure, which can execute appropriate reactive actions in order to advert SLA violations. In figure 2, we present a service architecture that builds on three main areas : agreement negotiation, brokering, and service deployment using virtualization. The architectures components are loosely coupled using SLAs

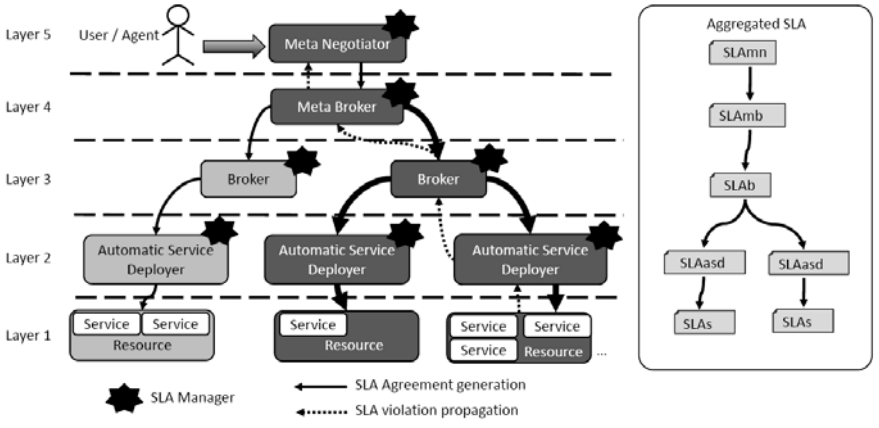


Fig. 2. Laysi Infrastructure

between the components. Thus, in case of failures components can be exchanged easily by renegotiating with another instance, e.g. another broker. The actors of the proposed architecture include users, Meta Negotiator [2], Meta Broker [8], Automatic Service Deployers (ASD) [9], services and physical resources.

The SLA negotiation is done like this: The User starts a negotiation for executing a service with certain QoS requirements. Then, the Meta negotiator asks the Meta broker, if it could execute the service with the specified requirements including required negotiation or security protocols. The Meta broker matches the requirements to the properties of the available Brokers and replies with an acceptance or a different offer for renegotiation. The aforementioned steps may continue for renegotiations until both sides agree on the terms (to be written to an SLA document) following the specific negotiation strategy or auction. Thereafter, the User calls the service with the Service Description (SD) and the agreed SLA. SDs describe a master image by means of a self-contained software stack (OS, middleware, applications, data, and configuration) that fully captures the functionality of the component type. Moreover, the SD contains information and rules necessary to automatically create service instances from a single parametrized master. Meta-negotiator passes the SD and the possibly transformed SLA (using a protocol the selected broker understands) to the Meta broker. The meta broker calls the selected Broker with the SLA and a possibly translated SD (to the language of the Broker). The Broker executes the service with respect to the terms of the SLA. The ASD monitors the states of the virtual resources and deploys services. SLA generation is done top-down as already described. The right hand side of Figure 2 shows the agreed SLAs (SLAmn for Meta negotiator SLA and so on).

Management of the SLA threat violation is done bottom-up on behalf of the SLA manager, which is implemented by each component of the SLA layered architecture [4]. The mechanism for the propagation of SLA violation threats is

based on Case Based Reasoning (CBR). If a layer is unable to cope with the violation threat, it propagates it to upper layers until a component on a a layer matches this threat with some historical case in its knowledge base and makes certain adjustments to avoid a possible SLA violation.

Lets take an example from the motivational scenario in section 2. The rendering workflow service finds out a possible violation threat with response time. Being unable to locate the problem, it propagates the violation threat to the Automatic Service Deployer layer (ASD). The ASD component looks for a similar case in its case history and finds out the root of the problem to be the latency caused by a drop in incoming bandwidth. The solution is also mentioned in the CBR history and that is to increase the incoming bandwidth by 10 per cent. Doing so speeds up the data migration from the user to the service and removes the violation threat.

3.3 Business-Level SLA Validation: A Rule-Based Validation System

Service value chains (as exemplified in the previous section) lead to a hierarchical structure of SLA contracts between different supply chain partners involving conflicting business concerns such as privacy trust and security etc. On the one hand it is not sensible to expose information of all the SLAs across the whole hierarchy of services as it will endanger the business processes creating added value, and on the other hand, for the sake of automated validation, certain information must be shared. To achieve this balance between privacy and trust we have introduced the concept of *SLA Views* [11],(see Figure 1(a)) an integral part of our rule based validation framework, where every service provider is restricted to its own view. Along the service value chains, SLAs are also aggregated in an incremental way [11]. The complete information of aggregated SLA at a certain level in the service chain is restricted to the corresponding SLA View and is only known by the service provider whereas only a filtered part is exposed to the immediate consumer. As the aggregation details are obscured at each level, a distributed top-down validation mechanism is a good strategy for the complete validation of a hierarchical SLA aggregation. The composed SLAs are required to be decomposed in an incremental manner down towards the supply chain of services and get validated in their corresponding service providers's domain. SLAs represented in form of rules contribute at the core of validation system.

We present an agent-enabled rule-based runtime validation framework for hierarchical SLAs, which allows the provisioning, delivery, and monitoring of services in service value chains as well as their highly dynamic and scalable consumption. The framework is based upon the Rule Responder Architecture [12], and the formal model of SLA Views [11]. Rule Responder [1] [12] is a rule-based enterprise service middleware for distributed rule inference services and intelligent rule-based Complex Event Processing on the Web and weaves the outer shell of the validation system by providing the required infrastructure for

[1] http://responder.ruleml.org

the automation of role description of partners as well as steering and redirection of the distributed validation queries.

Rule Responder adopts the approach of multi agent systems. There are three kinds of agents: (1) Organizational Agents (OA), (2) Personal Agents (PA), and External Agents (EA). A virtual organization is typically represented by an organizational agent and a set of associated individual or more specific organizational member agents. The organizational agent might act as a single agent towards other internal and external individual or organizational agents. In other words, a virtual organization's agent can be the single (or main) point of entry for communication with the "outer" world (external agents). Similar to an organizational agent, each individual agent (personal and external) is described by its syntactic resources of personal information about the agent, the semantic descriptions that annotate the information resources with metadata and describe the meaning with precise business vocabularies (ontologies) and a pragmatic behavioral decision layer which defines the rules for using the information resources and vocabularies/ontologies to support human agents in their decisions or react autonomously as automated agents/services. The flow of information is from External to Organizational to Personal Agent.

The aggregation of SLAs is a distributed mechanism and the aggregation information is scattered throughout the SLA choreography across various SLA views. To be able to validate the complete SLA aggregation, the validation query is required to traverse through all the SLA views lying across heterogeneous administrative domains and get validated locally at each SLA view. The terms of an SLA are represented as logical rules. These rules are composed together during the process of SLA aggregation [11]. The process of validation is performed by using these rules as distributed queries. During the validation process, queries are decomposed making their premises as subgoals. This backward chaining propagates throughout the SLA Choreography. If all the subgoals are satisfied then the validation is successful. The multi-agent architecture of Rule Responder provides communication middle-ware to the distributed stake-holders namely the client, the VOs and various service providers.

Let us discuss the scenario shown in 1(a). In the scenario, the user is interested to render her videos and then host them on the web. Her requirements include a maximum cost of 45 Euros, maximum response time of 5 seconds, minimum resolution of 1080X720 pixels and the minimum bandwidth (from hosting service) of 50 Mbps. In figure 3, we have depicted the same scenario from validation point of view. The user-requirements are shown on the top of the figure, expressed as a derivation rule composed of SLOs of the final aggregated SLA. The agents OA and PA representing the Rule Responder architecture, are shown to automate the distributed query processing. During the validation process, this rule will be decomposed such that each premise will become a subgoal. This subgoal will be sent as a message to the PA corresponding to the next SLA view in the hierarchy where it will emerge as the conclusion of one of the rules in the local rule set, thus forming a distributed rule chain. The initial steps of decomposition procedure are depicted at the bottom of the figure. For each service provider,

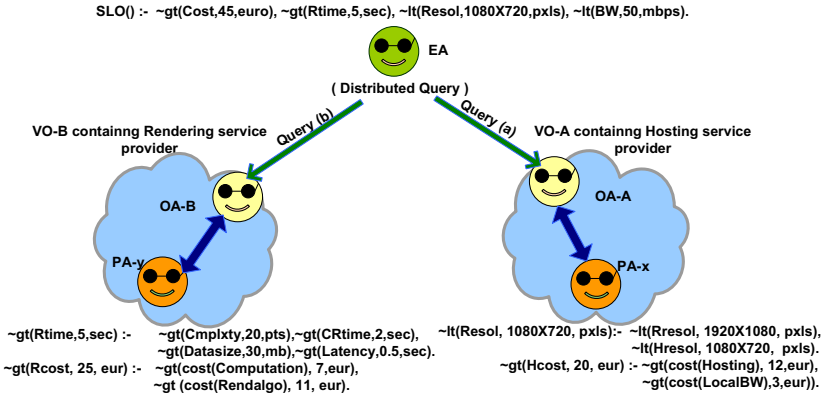


Fig. 3. Validation through Distributed Query Decomposition

there is a Personal Agent (PA). Service value chains are translated to PA-PA chains. The process continues until the query has found all the goals expressed in terms of logical rules. The true or false results are conveyed back following the same routes. At each level, the corresponding reward and penalty conditions are also checked and if required, appropriate action is taken. The distributed Rule Responder agent architecture acts as an enabling technology for the SLA Views concept.

4 A Holistic SLA Validation Framework for Layered Cloud Infrastructure

In the previous section we have described the role of SLAs and their validation within three different scopes i.e., infrastructure, business and resource levels and have explained the relevant validation mechanisms. Combining these three systems, we can come up with a holistic framework to populate, maintain and validate SLAs for service value chains in the Layered Cloud Infrastructures.

4.1 Functionality

The framework facilitates an agile formation of service value chains by utilizing the LAYSI infrastructure that contributes in SLA negotiation, brokering and service deployment. The holistic validation framework helps the service provider to prevent SLA violations. Every service provider has an instance of LoM2HiS monitoring system which is continuously keeping track of the SLA levels by mapping the low level resource metrics to the high level SLA parameters. Thereafter, the framework adjusts and tunes the supporting infrastructure continuously to sustain the expected quality of service. Using the LAYSI infrastructure, for proactive actions, information about possible violations is propagated to upper layers by the provider in a bottom up fashion. The corresponding components of the

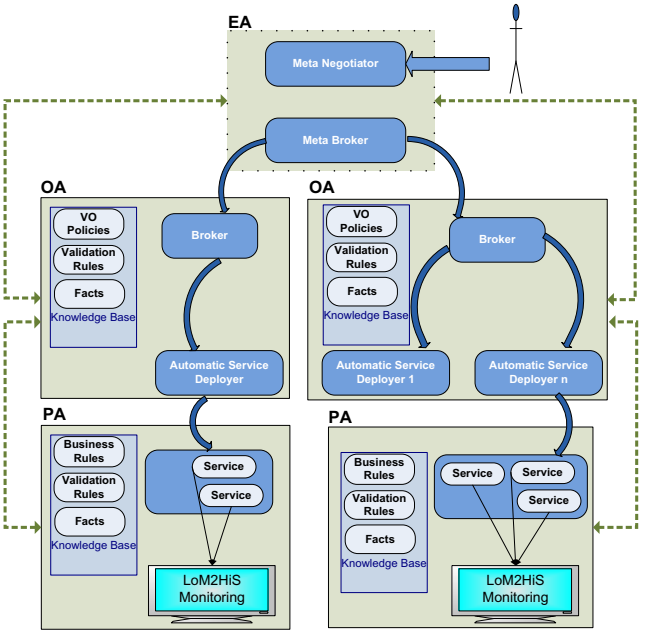


Fig. 4. A Holistic SLA Validation Framework

layer can adjust themselves by consulting a case based reasoning system as part of the LAYSI system. In addition to the historic system experiences necessary for the self management of the systems, we consider policies, validation rules and facts in for organizational agents and business rules, validation rules and facts for personal agents for utilizing the agent based validation system to validate the complaints initiated by consumers. User-driven SLA violation complaints in this case travel in top down manner to locate the violating body. Penalty enforcement can be executed once the violating party is found out.

4.2 Architecture

Fig 4 depicts an outline of the holistic SLA management framework. As we can see, three SLA management systems have been merged together to form a holistic SLA management framework. First two layers of LAYSI have been mapped on External Agent of the agent based validation system. The dotted rectangular shows that both layers are not always mapped together to EA. Broker and Automatics Service Deployer will always fall in the domain of an Organizational Agent. Every broker is mapped only on one Virtual Organization. A service provider in an organization is represented by its Personal Agent. A service provider may offer more than one services. It needs to monitor those service by observing the low level service metrics if they comply to the promised Service Level Objectives. For this purpose, an instance of the monitoring system of LoM2His will always be running within the domain of a service provider.

4.3 Description

Now we describe the behavior of the holistic validation framework with the help of the motivational scenario presented in figure 1(a). The LAYSI components cooperate closely with the agent based validation system. LAYSI helps to form new value networks and service value chains by helping in discovery, negotiation and maintenance of the services. When a client wants to find an atomic or composite service e.g., in our case the rendering workflow service and the hosting service, its request is translated from meta negotiator to the meta broker which depending upon the type of requirement contacts appropriate brokers. A broker represents a VO or a business enterprize. Services fulfilling client's requirements are discovered and present to the client. The negotiation and renegotiation processes are supported and facilitated by the mediating LAYSI components until the establishment of SLAs . The services shown in the service value chain of figure 1(a) go through the same discovery, (re)negoational process.

After the formation of SLAs through out the service value chain, LAYSI coupled with LoM2HiS system contributes to maintain the service QoS levels for all the services in conformance with the Service Level Objectives. LoM2HiS runs within a service provider's domain and keeps track of the possible SLA violation threats. Violation threats are produced whenever a service metric exceeds SLA violation threshold. Proactive actions are under taken immediately to bring the level of the service back to the permissible range. If the cause of the violation is beyond the scope of the service provider and is coming from upper layers then the violation is notified to the immediate upper layer. A possible violation cause for instance could be the incoming bandwidth which falls below a certain range causing the dependant processes to slow down. If the upper layer is responsible to control the level of the in coming bandwidth, then it can search for the possible solution by consulting its case-based-reasoning archive and finding an similar case from the history and looking for the corresponding solution. However, if the remedy is beyond the scope of this layer, the violation report will be notified to the layer above it. In this way, the SLA violations are propagated in a bottom up fashion for corrective measures.

Service validation queries initiated by clients are entertained by the Rule based SLA validation system. Following our example depicted in Fig 1, validation is initiated whenever meta negotiator, meta broker, broker or ASD changes the current SLA setup. It must be noticed that the violations travel in an opposite direction in the rule based validation system. LoM2HiS coupled with LAYSI work for the service provider to maintain the system and to take in time corrective measures in case of possible violation threats. Whereas the rule based validation system performs the validation test for two reasons: (i) a regular consistency check for the SLA aggregation corresponding to service value chain (ii) in case of SLA breach report thus initiating a sort of inquiry on client's behalf to locate the SLA violation point in the SLA aggregation and then to perform penalty enforcement.

5 Conclusion and Future Work

We have highlighted the importance of SLA validation from resource, infrastructure and business points of view by describing the corresponding SLA validation models. We have integrated the respective SLA validation models into a holistic validation framework. We have already started to integrate LoM2HiS and LAYSI systems into the FOSII framework and we are in the process of plugging in the Rule Based SLA validation framework as well.

References

1. Buyya, R., Yeo, C.S., Venugopal, S., Broberg, J., Brandic, I.: Cloud computing and emerging IT platforms: Vision Hype, and Reality for delivering computing as the 5th utility. Future Generation Computer Systems 25(6), 599–616 (2009)
2. Brandic, I., Music, D., Dustdar, S.: Service Mediation and Negotiation Bootstrapping as First Achievements Towards Self-adaptable Grid and Cloud Services. In: GMAC 2009 in conjunction with the 6th International Conference on Autonomic Computing and Communications Barcelona, Spain, June 15-19 (2009)
3. Maurer, M., Brandic, I., Emeakaroha, V.C., Dustdar, S.: Towards Knowledge Management in Self-adaptable Clouds. In: The proc. of SEASS 2010 in conjunction with ICWS 2010 and SCC 2010, Miami, USA (2010)
4. Brandic, I., Emeakaroha, V.C., Maurer, M., Acs, S., Kertész, A., Kecskeméti, G., Dustdar, S.: LAYSI: A Layered Approach for SLA-Violation Propagation in Self-manageable Cloud Infrastructures. In: CloudApp 2010, In conjunction with the 34th Annual IEEE International Computer Software and Applications Conference Seoul, Korea, July 19-23 (2010)
5. Emeakaroha, V.C., Brandic, I., Maurer, M., Dustdar, S.: Low Level Metrics to High Level SLAs - LoM2HiS framework: Bridging the gap between monitored metrics and SLA parameters in Cloud environments. In: The 2010 High Performance Computing and Simulation Conference (HPCS 2010), Caen, France, June 28- July 2 (2010)
6. Foundation of Self-governing ICT Infrastructures (FoSII), http://www.infosys.tuwien.ac.at/linksites/FOSII
7. Brein Project (Business objective driven reliable and intelligent Grids for real business) (2009), http://www.eu-brein.com
8. Kecskemeti, G., Kacsuk, K.: GMBS: A New Middleware Service for Making Grids Interoperable. Future Generation Computer Systems 26(4), 542–553 (2010)
9. Kecskemeti, G., Kacsuk, K.: GMBS: Automatic Service Deployment using Virtualization. In: PDP 2008, Toulouse, France (2008)
10. SLA@SOI (March 12, 2009), http://www.sla-at-soi.org/index.html
11. Haq, I.U., Huqqani, A., Schikuta, E.: Aggregating hierarchical service level agreements in business value networks. In: Dayal, U., Eder, J., Koehler, J., Reijers, H.A. (eds.) Business Process Management. LNCS, vol. 5701, pp. 176–192. Springer, Heidelberg (2009)
12. Paschke, A., Boley, H., Kozlenkov, A., Craig, B.: Rule responder: RuleML-based agents for distributed collaboration on the pragmatic web. In: Proceedings of the 2nd international conference on Pragmatic web Tilburg, The Netherlands (2007)
13. Haq, I.U., Paschke, A., Schikuta, E., Boley, H.: Rule-Based Workflow Validation of Hierarchical Service Level Agreements. In: The 4th International Worshop on Workflow Management, Geneva, Switzerland (2009)

Author Index