BIRKHÄUSER

AUTONOMIC SYSTEMS

The AUTONOMIC SYSTEMS book series provides a platform of communication between academia and industry by publishing research monographs, outstanding PhD theses, and peer-reviewed compiled contributions on the latest developments in the field of autonomic systems.

It covers a broad range of topics from the theory of autonomic systems that are researched by academia and industry. Hence, cutting-edge research, prototypical case studies, as well as industrial applications are in the focus of this book series. Fast reviewing provides a most convenient way to publish latest results in this rapid moving research area.

The topics covered by the series include (among others):

- self-* properties in autonomic systems (e.g. self-management, self-healing)
- architectures, models, and languages for building autonomic systems
- trust, negotiation, and risk management in autonomic systems
- theoretical foundations of autonomic systems
- applications and novel computing paradigms of autonomic systems

Economic Models and Algorithms for Distributed Systems

Dirk Neumann
Mark Baker
Jörn Altmann
Omer F. Rana
Editors

Birkhäuser
Basel · Boston · Berlin

Editors:

Dirk Neumann
Chair for Information Systems
Kollegiengebäude II
Platz der Alten Synagoge
79085 Freiburg
Germany
e-mail: dirk.neumann@is.uni-freiburg.de

Jörn Altmann
Technology Management, Economics
& Policy Program
College of Engineering
Seoul National University
San 56-1, Shillim-Dong,
Gwanak-Gu, Seoul 151-742
South Korea
e-mail: jorn.altmann@acm.org

Mark Baker
Research Professor of Computer Science
ACET Centre, School of Systems Engineering
The University of Reading
Whiteknights, Reading
Berkshire RG6 6AY
UK
e-mail: mark.baker@computer.org

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

1998 ACM Computing Classification: C.2.4 [Distributed Systems]; C.2.1 [Network Architecture and Design]: Distributed networks: Network communications; C.2.3 [Network Operations]; C.4 [Performance of Systems]; H.3.4 [Systems and Software]: Distributed systems; I.2.11 [Distributed Artificial Intelligence]; K.6.4 System Management

Library of Congress Control Number: 2009931265

Bibliographic information published by Die Deutsche Bibliothek.
Die Deutsche Bibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data is available in the Internet at http://dnb.ddb.de

ISBN 978-3-7643-8896-6 Birkhäuser Verlag AG, Basel – Boston – Berlin

ISBN 978-3-7643-8896-6 ISBN 978-3-7643-8899-7 (eBook)

9 8 7 6 5 4 3 2 1 www.birkhauser.ch

Contents

Economic Models and Algorithms for Distributed Systems, 1–3
Book Series: Autonomic Systems

Economic Models and Algorithms for Distributed Systems

Modern computing paradigms have frequently adopted concepts from distributed systems. The quest for scalability, reliability and cost reduction has led to the development of massively distributed systems, which extend organisational boundaries. Voluntary computing environments (such as BOINC), Grids (such as EGEE and Globus), and more recently Cloud Computing (both open source and commercial) have established themselves as a range of distributed systems.

Associated with this advance towards cooperative computing, the paradigm of software agents generally assumes that cooperation is achieved through the use of obedient agents that are under centralised control. In modern distributed systems, this main assumption is no longer valid. On the contrary, cooperation of all agents or computing components is often necessary to maintain the operation of any kind in a distributed system. Computer scientists have often considered the idea that the components of the distributed system are pursuing other selfish objectives, other than those that the system designer had initially in mind, when implementing the system. The peer-to-peer file sharing systems, such as BitTorrent and Gnutella, epitomise this conflict of interest, because as low as 20% of the participants contribute more than 80% of the files. Interestingly, various distributed systems experience different usage patterns. While voluntary computing environments prospered through the donation of idle computing power, cooperative systems such as Grids suffer due to limited contribution from their participants. Apparently, the incentive structure used to contribute to these systems can be perceived differently by the participants.

Economists have also demonstrated research interest in distributed systems, exploring incentive mechanisms and systems, pioneered by Nobel-prize winners von Hayek and Hurwicz in the area of incentives and market-based systems. As distributed systems obviously raise many incentive problems, economics help complement computer science approaches. More specifically, economics explores situations where there is a gap between individual utility maximising behaviour and socially desirable deeds. An incorrect balance between such (often conflicting) objects could lead to malfunctioning of an entire system. Especially, cooperative computing environments rely on the contribution of their participants. Research test beds such as EGEE and PlanetLab impose regulations on the participants

that contribute, but the enforcement of these institutions is informal by the loss of reputation.

While such a system is dependent on the reputation of the participants that work in academia, a commercial uptake has been limited. In the past, it became evident that cooperative computing environments need incentive mechanisms that reward contribution and punish free-riding behaviour. Interestingly, research on incentive mechanisms in distributed systems started out in economics and computer science as separate research streams. Early pioneers in computer science used very simple incentive mechanisms in order to align individual behaviour with the socially desirable deeds. The emphasis was on the implementation of these mechanisms in running computing environments. While these studies demonstrate that it is possible to combine the principles of economics in sophisticated (Grid) middleware, it has also become evident that the mechanisms were too simple to overcome the effects of selfish individual behaviour. Interestingly, research in economics pursued a diametrically opposing approach. Abstracting from the technical details of the computing environments, were sophisticated mechanisms were developed that demonstrated desirable economic properties. However, due to the abstract nature of these mechanisms a direct implementation is not always possible.

It is, nevertheless, interesting to see that these initially different research streams have been growing together in a truly inter-disciplinary manner. While economists have improved their understanding of overall system design, many computer scientists have transformed into game theory experts. This amalgamation of research streams has produced workable solutions for addressing the incentive problems in distributed systems.

This edited book contains a compilation of the most recent developments of economic models and algorithms in distributed systems research. The papers were selected from two different workshops related to economic aspects in distributed systems, which were co-located with the IEEE Grid 2007 conference in Austin and with the ACM MardiGras 2008 conference in Baton Rouge. The extended papers from these events have been added to by projects being funded by the European Union, which in particular, address economic issues in Grid systems. As Grid computing has evolved towards the use of Cloud infrastructure, the developed economic algorithms and models can similarly be utilised in this new context – in addition to further use within peer-to-peer systems.

This book inevitably emphasises computing services, which look at the economic issues associated with contracting out and the delivery of computing services. At the outset of each service delivery the question arises, which service request will be accommodated at what price, or is it even provided free of charge. As these issues are spawned around business models and in particular around markets as a special kind of business model, the first chapter is devoted to the exploration of these questions. Once it has been determined, in order to resolve which service request should be accepted, a formal contract needs to be defined

and mutually signed between service requester and provider. The second chapter of the book deals with aspects of service-level agreements (SLAs). One particular emphasis is on how infrastructure providers (e.g. Cloud vendors) maximise their profit, such that the Quality of Service (QoS) assertions specified in the SLA are always maintained. In the last phase of the transaction chain stands the enforcement of the SLAs. In case of detected SLA infringements (which may be by the client or the provider, but with a focus generally on the provider), penalty payments will be need to be paid by the violating provider. If the services are small-scale, it is in many cases too costly to enforce penalty payments by law. Thus, there is a need to enforce the SLAs without formal legal action; otherwise the contracts would prove to be worthless. A current practice is to establish trust among the service providers by means of reputation systems. Reputation systems embody an informal enforcement, where the SLA violators are not punished by the requester, whose SLA was breached, but by the community, which may subsequently limit use of the service offering from the respective provider. The design of reputation mechanisms is often quite difficult to undertake in practice, as it should reflect the actual potency of a provider and not be politically motivated.

Part I:
Reputation Mechanisms and Trust

Economic Models and Algorithms for Distributed Systems, 7–8
Book Series: Autonomic Systems

Reputation Mechanisms and Trust

Reputation mechanisms and trust as well as Service Level Agreements, addressed in the previous section are somewhat complementary. Whereas SLAs primarily encode contractual obligations between consumers and providers, reputation models enable choice of providers based on their past performance (assuming provider identity is persistent or traceable), or on their ability to deliver on these contractual obligations over time. Where "trust" is often defined between two participants, "reputation" often involves aggregating views from a number of different sources.

It is useful to note that when developing reputation mechanisms, not all aspects (i.e. capabilities offered by a provider) need to be considered as part of the reputation model – hence, depending on the context of usage, reputation may be calculated differently. This forms the basis for the reputation model from Ali and Rana in their chapter "*Belief-based Trust Model for Dynamic Service Selection*", where reputation is calculated based on the particular context of use, or subjective belief of a participant. The authors attempt to combine various views on reputation and trust, depending on how these terms are perceived by a user. They subsequently demonstrate how trust may be used as a selection criterion between multiple service providers.

Anandasivam and Neumann continue this theme in their chapter "*Reputation, Pricing and the E-Science Grid*" by focusing on how the use of reputation can be used to incentivise a provider, essentially preventing such a provider from terminating a computational job from a client, even though the provider could make greater revenue by running an alternative computational job. Their work compares job submission with sites that do (and do not) use reputation mechanisms, and discuss how price determination can be associated with reputation – and present the associated decision model that may be used by market participants. Most importantly, they demonstrate that the correct use of price setting enables better collaborative interactions between participants.

The next two chapters focus on the formation of communities and virtual organizations in order to allow participants to maximise their reward (or "utility"). Kastidou and Cohen in their chapter "*Trust-oriented Utility-based Community Structure in Multiagent Systems*" discuss how better community structures could be established by allowing their participants to exchange reputation information. In this way, reputation may serve as either an incentive or a barrier to entry for an agent attempting to join another community. The focus of their work is

on the incentive mechanisms for communities to truthfully and accurately reveal reputation information, and the associated privacy concerns about disclosing such information to others. Their work is particularly relevant in open environments, as exemplified through file sharing Peer-2-Peer systems, where a decision about what files to share (upload/download) and from which participants, becomes significant.

The chapter from Carroll and Grosu entitled "*Formation of Virtual Organizations in Grids: A Game-Theoretic Approach*", has a similar focus. They consider the formation of Virtual Organizations (VOs) which involves the aggregation of capacity from various service providers –which has a similar scope, although a different focus (on application/job execution, rather than community structure) to the notion of communities in the chapter by Kastidou and Cohen. They discuss incentive mechanisms that would enable self interested Grid Service Providers (GSPs) to come together to form such VOs using a coalitional game-theoretic framework. They demonstrate how given a deadline and a budget, VOs can form to execute particular jobs, and then dissolve. They use Myerson's cooperation structure to achieve this, and rely on the assumption that GSPs exhibit welfare maximising behaviours when participating in a VO.

A last chapter in this section looks more at the payment issue emphasizing the business perspective of cooperative computing infrastructures. The paper "*Towards Dynamic Authentication in the Grid -Secure and Mobile Business Workflows using GSet*" by Mangler, Schikuta, Witzany, Jorns, Ul Haq and Wanek introduce the use of gSET (Gridified Secure Electronic Transaction) as a basic technology for trust management and secure accounting in cooperative computing environments.

Economic Models and Algorithms for Distributed Systems, 9–23
Book Series: Autonomic Systems

A Belief-based Trust Model for Dynamic Service Selection

Ali Shaikh Ali and Omer F. Rana

Abstract. Provision of services across institutional boundaries has become an active research area. Many such services encode access to computational and data resources (comprising single machines to computational clusters). Such services can also be informational, and integrate different resources within an institution. Consequently, we envision a service rich environment in the future, where service consumers can intelligently decide between which services to select. If interaction between service providers/users is automated, it is necessary for these service clients to be able to automatically chose between a set of equivalent (or similar) services. In such a scenario trust serves as a benchmark to differentiate between service providers. One might therefore prioritize potential cooperative partners based on the established trust. Although many approaches exist in literature about trust between online communities, the exact nature of trust for multi-institutional service sharing remains undefined. Therefore, the concept of trust suffers from an imperfect understanding, a plethora of definitions, and informal use in the literature. We present a formalism for describing trust within multi-institutional service sharing, and provide an implementation of this; enabling the agent to make trust-based decision. We evaluate our formalism through simulation.

1. Introduction

The existence of online services facilitates a novel form of communication between individuals and institutions, supporting flexible work patterns and making an institutional's boundaries more permeable. Upcoming standards for the description and advertisement of, as well as the interaction with and the collaboration between on-line services promise a seamless integration of business processes, applications, and online services over the Internet. As a consequence of the rapid growth of on-line services, the issue of trust becomes significant. There are no accepted

techniques or tools for specification and reasoning about trust. There is a need for a high-level, abstract way of specifying and managing trust, which can be easily integrated into applications and used on any platform. The need for a trust-based decision becomes apparent when service consumers are faced with the inevitability of selecting the *right* service in a particular context. This assumes that there is likely to be a service-rich environment (i.e. a large number of service providers) offering similar types of services. The distributed nature of these services across multiple domains and organizations, not all of which may be trusted to the same extent, makes the decision of selecting the *right* service a demanding concern, especially if the selection proves is to be automated and performed by an intelligent agent.

We present a formalized approach to manage trust in online services. Our work contributes the following to the research in this field: (1) a detailed analysis of the meaning of trust and its components; (2) a trust model based on a socio-cognitive approach; (3) a trust adaptation approach; (4) an approach for service selection based on trust (using different criteria). The remainder of this article is structured as follows. First, we provide an overview of related work (Section 3.). We then present a brief overview of methodology we apply for deriving the formalism, in Section 4.. In Section 5. a discussion of the trust system and its components is presented. In Section 7. we present our approach, and the evaluate it in Section 8..

2. Motivations

In order to exemplify our trust formalism we will apply it to a particular scenario, based on the Faehim (Federated Analysis Environment for Heterogeneous Intelligent Mining) toolkit [8]. The aim of the Faehim project is to develop machine learning Web Services and combine them using the Triana workflow engine for Web Services composition. The scenario involves a user confronted with the inevitability of selecting a machine learning Web Service within the workflow. The potential number of suitable services is large, and services are deployed with different qualities, i.e. speed, reliability, etc. The scenario makes use of multiple such services (such as a regression technique, a clustering technique, etc). In such a scenario, the user should make a trust-based selection that enables service prioritization based on their beliefs about service quality. It is intended that the user should select a service that most matches his trust preferences or policy.

3. Related work

The general notion of trust is excessively complex and appears to have many different meanings depending on how it is used. There is also no consensus in the computer and information sciences literature on what trust is, although its

importance has been widely recognized and the literature available on trust is substantial. Broadly speaking, there are two main approaches to trust introduced in the literature. The first approach aims to allow agents to trust each other and therefore there is a need to endow them with the ability to reason about the reliability or honesty of their counterparts. This ability is captured through trust models. The latter aim to enable agents to calculate the amount of trust they can place in their interaction partners. This is achieved by guiding agents on decision making in deciding on how, when and who to interact with. An agent in this context refers to either a service user or a provider. However, in order to do so, trust models initially require agents to gather some knowledge about their counterparts. This has been achieved in three ways in the literature:

1. **A Presumption drawn from the agent's own experience:** Trust is computed as a rating of the level of performance of the trustee. The trustee's performance is assessed over multiple interactions to check how good and consistent it is at doing what it says it does. To this end, Witkowski et al. [10] propose a model whereby the trust in an agent is calculated based on its performance in past interactions. Similar to Witkowski et al., Sabater et al. [9] (using the REGRET system) propose a similar model but do not just limit the overall performance to the agent's direct perception, but they also evaluate its behavior with other agents in the system.
2. **Information gathered from other agents:** Trust in this approach is drawn indirectly from recommendations provided by others. As the recommendations could be unreliable, the agent must be able to reason about the recommendations gathered from other agents. The latter is achieved in different ways: (1) deploying rules to enable the agents to decide which other agents' recommendation they trust more, as introduced by Abdul-Rahman et al. [1]; (2) weighting the recommendation by the trust the agent has in the recommender, EigenTrust [5] and PageRank [7] are examples of this approach.
3. **Socio-cognitive trust:** Trust is drawn by characterizing the known motivations of the other agents. This involves forming coherent beliefs about different characteristics of these agents and reasoning about these beliefs in order to decide how much trust should be put in them. An example of this is work by Castelfranchi [3].

While trust models pertain to the reasoning and information gathering ability of agents, the second main approach to trust concerns the design of protocols and mechanism of interaction. A common protocol for interaction is user authentication – a common technique found in many types of applications. This involves a username/password combination to access a service. A common variation to this technique is the use of Digital Certificates to verify a user's identity. This verification is done through a third agent that creates a unique encrypted certificate for every user machine. The certificate is then submitted when the user makes a request to another agent.

4. The methodology

A difficult issue when discussing trust is that the phenomenon is such a subjective one [6]. It is difficult to provide a comprehensive definition for trust, and is the reason why previous studies have failed to provide a detailed definition. Previous projects have therefore restricted themselves to just one or two of the several aspects of trust. In our approach, instead of starting with a definition and developing a formalism, we start with intuitive ideas about how trust works by breaking down the components of trust, coupled with investigation of how these components may be aggregated to support a *trust decision*, and attempting to develop a formalism around this decision process. The advantage of this approach is that we will cover more or less many aspects of trust in the formalism. In addition, having studied the components of trust, we could make use existing literature to derive the formalism.

5. Trust components

Currently there is no consensus on a precise definition of trust nor its basic components. However, there is a general agreement on the subjective nature of trust. Consider for example the definition of trust provided by Gambetta [4]: "Trust is a particular level of the *subjective probability* with which an agent assesses that another agent will perform a particular action." The definition stresses that trust is basically an estimation, an opinion and an evaluation. Similarly, the Oxford Reference Dictionary defines trust as a belief: "Trust is the firm *belief* in the reliability or truth or strength of an entity." Trust can also be related to the role undertaken by an individual in a particular context, based on the specific goals and priorities of that role. Essentially therefore, trust means different things to different people, to different roles, and in different scenarios. Trust can mean such things as the following:

- Do I believe that what someone says is true and factual?
- Do I agree with a person or an organisation's goal, or what they stand for?
- Do I believe that a person or organisation's goal(s) and/or priorities match mine?

The above discussion leads us to draw an explicit terminology for trust. As our intention is to allow a client to make a trust-based decision for selecting service providers, we specify trust as an assumption or an expectation we make about others in some context/environment. This expectation is based upon more specific beliefs which form the basis or the components of trust [3]. These are beliefs relating to a provider that a client wishes to trust. Such beliefs are the answer for the question: "What do we have in mind when we trust a service?" For example, we may trust a service because we *believe* that service is able to do what we need (competence) and it will actually do it quickly (promptness). Competence and

promptness are therefore examples of the basic belief and mental state components of trust, in this instance. Hence, the importance of any particular criteria is dependent on the client making use of a service. Some clients may be interested in promptness, and others in accuracy. We therefore take account of the fact that trust values may need to be defined with reference to different criteria. We may classify beliefs according to the context of service provision into the following:

1. **Non-situational beliefs:** These beliefs concern the trustee, and do not relate to the currently on-going transaction. Institutional beliefs include:
    - Competence Belief: the ability of a service provider to accomplish a task, such as providing accurate results or performing a desired action [3].
    - Availability Belief: a belief that the service will be on-line and available when a request to it is sent.
    - Promptness Belief: The speed at which the service responds to task requests by accomplishing the agreed upon task.
    - Cost Belief: Cost refers to the monetary value that the user is willing to pay.
2. **Situational beliefs:** These beliefs concern the situation of the truster and the benefit that he will get from the trusting decision. Situational beliefs include:
    - Harmfulness Belief: These beliefs concern the risks of propagating a task (or data) for execution (or processing) to a given service provider.
    - Importance Belief: These beliefs concern the user-centered judgment of the importance of the task. The greater the importance, the greater the likelihood to trust. This indicates that if accomplishment of a particular task is significant to a client, there will be a greater importance in the need to trust the service provider.
    - Utility Belief: Utility refers to the benefits that the user will gain from the task being successfully completed.

Several benefits can be derived from exploring the various types of beliefs. A client may therefore prioritize potential service providers based on evaluating the beliefs outlined above. For example, if a client knows that a goal must be achieved quickly, even at the price of reduced accuracy, then the client might rank Web Services based on availability and promptness, with less concern for accuracy – such as intent or competence. If however, the goal must be accomplished with exact correctness, intent and competence, then these beliefs take precedence in the prioritization of Web services.

5.1 The sources of beliefs

Beliefs can come from two sources: the direct experience of a client, or as an acceptance of recommendations made by other clients of a particular provider. We classify the sources of beliefs into:

1. **Self-generated belief:** Self-generated beliefs are those that a client creates itself.
    - **Direct experience:** this means trying things out in practice, observing things and generally getting obtaining suitable evidence before committing to a belief. In reality, a client may only have time to try a limited number of operations.
2. **Externally generated belief:** The alternative to generating beliefs through a client is to utilize comments generated by other clients. Sources in this category include:
    - **Recommendations:** these constitute a form of advice from another client or organization (and are weighted by the trust placed in the recommender itself).
    - **Reputation:** this is a term that lacks a widely accepted definition. In our framework we define it as how other users feels about the behavior of a particular service provider. This may constitute an aggregate of views from multiple users about a single provider.

6. Illustrating beliefs

We present beliefs as a diagram – as a second step towards a formalism. From Figure 1, beliefs are represented by a circle where the circle indicates the type of the belief. The sources of the beliefs are represented by rectangles. The value that the source creates is written on the arrow. We also propose a weight for the source and present it as a small square at the top left of the source's rectangle. The weight value indicates how much we rely on that source. More about weights in Section 7.2.

7. Deriving a trust formalism

In this section we outline how a trust formalism may be derived using the concepts discussed in previous sections.

7.1 Combining belief values from various sources

The value of a belief should reflect the accumulation of all values produced by various sources, combined with the uncertainty associated with the nature of these sources. Two issues should be considered: (1) how the belief values can be combined, and (2) how do we deal with the uncertain nature of the belief sources. For the first issue, Castelfranchi et al. [3] propose an implementation for computing the trust value for the socio-cognitive model using Fuzzy Cognitive Maps (FCM). An FCM is an additive fuzzy system with feedback; it is well suited for

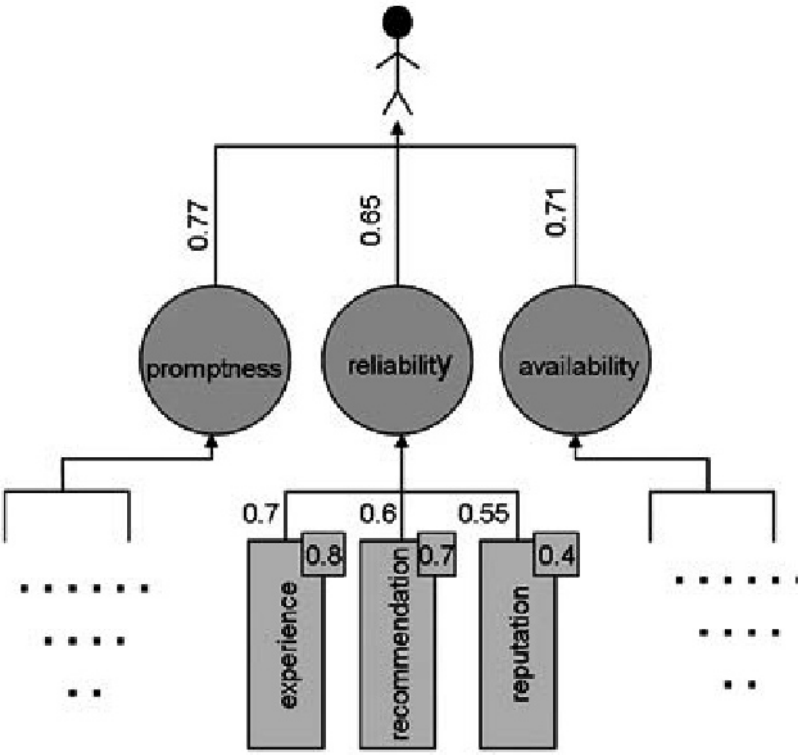

FIGURE 1. A complete scenario.

representing a dynamic system with cause-effect relations. An FCM has several nodes; representing belief sources, and edges, representing the *casual power* of a node over another one. The values of all the edges are assigned by a human and propagate in the FCM until a stable state is reached; so the values of the other nodes are computed. Two main problems are deduced from the FCM approach: (1) FCM does not take the uncertainty associated with sources into consideration, and (2) FCM assumes a human interaction to assign the value to the edges, which is limiting if the aim is to automate the trust decision.

It is possible to characterize the uncertainty associated with a given belief using a probability measure. However, the recent criticisms of the probabilistic characterization of uncertainty claim that traditional probability theory is not capable of capturing subjective uncertainty. The application of traditional probabilistic methods to subjective uncertainty often utilizes Bayesian probability. An additional assumption in classical probability is entailed by the axiom of additivity, where all probabilities that satisfy specific properties must add to 1. This forces the conclusion that knowledge of an event necessarily entails knowledge of

the complement of an event, i.e., knowledge of the probability of the likelihood of the occurrence of an event can be translated into the knowledge of the likelihood of that event not occurring.

As a consequence of these concerns, many more general representation of uncertainty to cope with particular situations involving uncertainty have been proposed. Dempster–Shafer Theory (DST) is a theory that was developed to cope with such particular situation. We use DST to combine trusting beliefs. The DST will be applied on all the beliefs obtained from the various sources.

7.2 Weighted Dempster–Shafer theory

Based on standard Dempster–Shafer theory, let the universal set be donated θ. Elements of θ represents mutually exclusive hypothesis. In our case, these elements represent one of the core beliefs in trust, i.e. competence, promptness, etc. With the universe of discernment θ defined, each source S_i would contribute its observation by assigning its belief values over θ. This assignment function is called the basic probability assignment (BPA) of S_i, denoted m_i. Formally, one defines BPA as the mapping, $m : 2^\theta \rightarrow [0, 1]$ that satisfies

$$\sum_{A \subseteq \theta} m(A) = 1\,.$$

Often, $m(\phi) = 0$, where ϕ is the null set. The belief in a subset $B \subset \theta$ is then defined as

$$bel(A) = \sum_{B \subseteq A} m(B)\,.$$

This indicates that belief in A can also be characterised with respect to a subset of A. DST assumes practical relevance since it is possible to revise the estimates based on information that may be available from additional (independent) sources. Suppose, for example that the estimate from one source is denoted by $m_1(A)$ and that from the other sources is denoted as $m_2(A)$. Dempster's rule of combination provides a belief function based on the combined evidence. The conjunctive rule of combination handles the case where both sources of information are fully reliable. The result of the combination is a joint BPA representing the conjunction of the two pieces of evidence induced from the two sources. This rule is defined as

$$(m_1 \cap m_2)(A) = \sum_{B, B \subseteq \theta, B \cap C = A} m_1(B) m_2(C)\,.$$

This is the un-normalized Dempster's rule of combination. If necessary, the normality assumption may be recovered by dividing each value by a normalisation coefficient:

$$(m_1 \oplus m_2)(A) = \frac{(m_1 \cap m_2)(A)}{a - m(\phi)}\,, \quad \forall \phi \neq A \subseteq \theta$$

where the quantity $m(\phi)$ is called the degree of conflict between m_1 and m_2 and can be computed using

$$m(\phi) = (m_1 \cap m_2)(\phi) = \sum_{B \cap C = \phi} m_1(B) m_2(C) \,.$$

The fundamental DST combination rule implies that we trust any two sources S_i and S_j equally. However, these sources are not always reliable and we usually trust some sources more than others, i.e. one might have greater belief values from ones own experience than belief values from received recommendations. This sort of deferential trust can be accounted for by a simple modification to DST, in which the observations m_i are weighted by trust factors w_i derived from the corresponding expectations, histories of the corresponding source S_i's performance. The weighting process has already been investigated by Basak et al. [2]. Their proposed formula of weighted DMS is defined as follows:

$$m_1 \triangle m_2 = m_1 \cap m_2$$

where $\triangle$ denotes the combination with usual Dempster's rule and

$$m_i = \frac{m_i^{wi}(A)}{\sum_{B \subseteq \theta} m_i^{wi}(B)} \,.$$

7.3 Trust adaptation: Dynamic weighting

When the ground truth is available, e.g. shortly after current measurements or from additional information channels, it can be used by making the weight factors w_i as functions of time. A simple but effective practical implementation of such an approach is to define:

$$w_i = \sum_{n=0}^{\infty} c_i(n).\frac{1}{n}$$

where $c_i(n)$ is the function describing the correctness of source S_i's estimation at time n:

$$c_i(n) = \begin{cases} 0 & \text{correct estimation} \\ 1 & \text{incorrect estimation} \end{cases}$$

and the $\frac{1}{n}$ is the "penalty factor", which is used to control the changes of the weight value. We aim from this penalty factor to:

- Increase the weight of the source by a large value if it gives correct estimation at the first stages and vice versa.
- Increase the weight of the source by a small value at later stages.

7.4 Trust computation and selection

Using weighted DST, it is not possible to compute the aggregated values for a belief from different independent sources. The next step is to aggregate the beliefs to derive a single "trust value". The trust value forms a benchmark for selecting services. In our case, the service that has the highest trust value is selected. Each belief influences the trust value and is associated with an influence factor k. This value indicates how a belief influences the eventual trust decision. The value of k is either positive or negative in the range [-1..1], such that:

$$w : \begin{cases} >= 0 & \text{when the belief promotes the trust value} \\ < 0 & \text{when the belief inhabits the trust value}. \end{cases}$$

For example, the k for the promptness belief might be assigned a positive value as it promotes trust, whereas the k for the harmfulness belief might be assigned a negative value as it inhabits trust. The trust value is computed as the sum of all the influencing beliefs:

$$trustvalue = \sum_{i=0}^{n} k_i * belief_i$$

where n is the number of the influencing beliefs, k_i is the weight of the $belief_i$.

8. Empirical evaluation

The primary goal of our evaluation is to show empirically that the formalism works and enables the system to make a service selection based on a trust-decision. The experiment is made up of a series of simulations. We first give an overview of the environment for the simulations and then discuss the expected and actual results followed by a discussion of the results.

8.1 Environment overview

We construct a simulation that allows the creation of different types of consumers in Java. Each consumer can have its own trust preference or policy. The simulation also allows the creation of user-specified belief sources, e.g. reputation, experience, etc. The following simulation parameters can also be specified:

- Service Quality Adjustments: these are runtime behavioural modifications to a service, specified to affect the execution performance of a service.
- Belief Source Adjustments: these are runtime behavioural modifications to a belief source, specified to affect the accuracy of the replicated belief values.

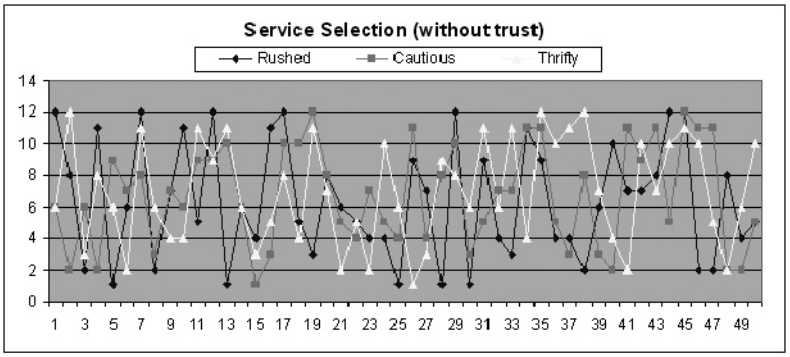

FIGURE 2. Simulation 1.

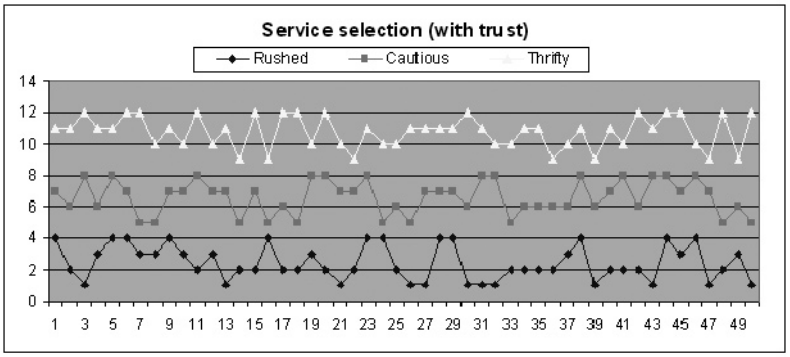

FIGURE 3. Simulation 2.

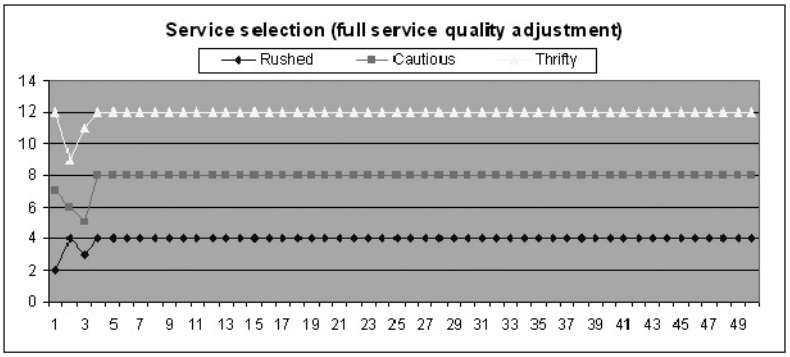

FIGURE 4. Simulation 3.

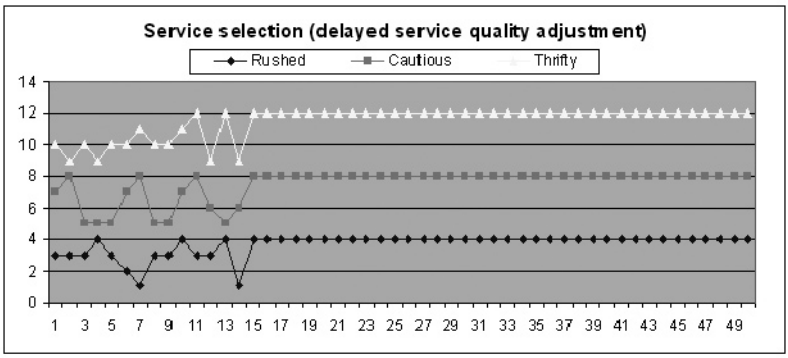

FIGURE 5. Simulation 4.

8.2 Setup summary

We conducted four simulations for this experiment. Each simulation consists of three groups of four identical machine learning services and three consumers. These machine learning services are based on the Faehim toolkit [8]. The machine learning services implement the J48 machine learning algorithm – an implementation of C4.5, a standard algorithm that is widely used for practical machine learning producing decision tree models. This algorithm works by forming pruned partial decision trees (built using C4.5's heuristics), and immediately converting them into a corresponding rule [11]. The Web services are deployed on an Apache/Axis server installed on a Windows platform. The services has one main operation: *classify* which takes in the input a file containing the data set, and returns a string representing the `J48` decision tree.

We implement each simulation using services with predictable behaviors. The primary service domain is Machine Learning. The domain and service interface are kept simple to facilitate measurement and fine-tuning of system parameters. The following artifacts are used in the experiment.

- Service Consumers. We deploy three types of consumers, each with its own trust policy:
  1. Cautious: as the name implies, this consumer's primary concern is safety.
  2. Thrifty: this consumer's policy is to primarily find any low cost service.
  3. Rushed: this consumer is primarily concerned with execution speed.
- Service Quality Adjustment. A service is adjusted by artificially deceasing or increasing a particular quality. For this experiment we introduce three types of service adjustments:
  1. DelayAdjustment: introduces a delay in a service method invocation.

2. FaultAdjustment: Increases a service method fault rate by introducing artificial faults.
3. CostAdjustment: Increases a service cost artificially.

- Belief Sources. We deploy two types of belief sources: reputation and experience sources. For the purpose of this experiment, we make these sources trustworthy. That is, these sources will always give correct estimations about the behaviour of the services.

8.3 Results

For each simulation we show the obtained results to illustrate the service selection choice for each type of customers. For each graph, the y-axis denotes the service number of the selection. Services are numbered according to table below. The x-axis shows the execution sequence for the consumers.

Service pool	Size	Adjusted	Clean
Fast pool	4	$\{1, 2, 3\}$	$\{4\}$
Reliable pool	4	$\{5, 6, 7\}$	$\{8\}$
Economic pool	4	$\{9, 10, 11\}$	$\{12\}$

8.3.1 Simulation 1: Service selection without trust

In this simulation we run the simulation without using the trust formalism for the entire duration of the simulation. That is, the service consumers do not consider the aggregated beliefs in their selection decision.

As expected, the results in Figure 2 show that consumers randomly select between services. The remaining simulations show what happens when the consumers start enabling the trust formalism in their selection decision.

8.3.2 Simulation 2: Service selection with trust

This time we run the simulation taking account of the trust formalism. Since we do not adjust the quality of the services of each pool, we would expect that the consumer would randomly select a service from the service pool, based on specification in their policy. Figure 3 shows that for all the three pools of consumers, the service selections are as expected.

8.3.3 Simulation 3: Full service adjustment

In this simulation all services but the last numbered service of each pool are adjusted negatively. That is, for the "Fast" pool we add a delay in the execution speed, for the "Reliable" pool we artificially increase the service method fault rate, and for the "Economical" pool we artificially increase the cost of the services. Since the last service of each pool is clean, we would expect the consumer would, in time,

find the service and increasingly select it. Figure 4 shows that for the the pools of consumers, we obtain convergence of all consumers for each pool to the clean service instance of each service pool.

8.3.4 Simulation 4: Delayed service adjustment

In this simulation we introduce a delay for all adjustments. Essentially, all service adjustment will only start occurring after the 10^{th} invocation for a particular service. Figure 5 shows the obtained results. The delay essentially shifts the convergence to the clean service to the right of the graph.

8.4 Discussion

In the previous section we have presented the results we obtained from our simulations. Based on the results, we observe a major advantage for using the formalism introduced previously. The results show how the use of trust can be used effectively to chose between the available services. Using such metrics allows for better overall decision making capability. The system became more intelligent and has the capability to utilize trust beliefs of various sources when making a service selection. Another advantage is that the system keeps watching the behavior of each service, and takes it into consideration when a service start behaving erroneously.

9. Conclusion and future work

The rapid growth of online services indicates that on-line communities should be able to make a trust-based decision to chose between services. In this paper, we introduced a formalism for trust that can be embedded in an intelligent agent: enabling it to make a trust-based decision. We derived our formalism by adapting a methodology based on the weighted Dempster–Shafer Theory. Using this approach, we investigate the various components of trust, and show how these components are aggregated to form a trust decision. We also introduce a trust adaptation approach by dynamically changing the weight of the sources based on their historical performance. We evaluate our finding by several simulations and discuss the advantages of the formalism. In future work we aim to consider composite services in a workflow and look on how the formalism could be applied within the Triana workflow engine.

References

[1] A. Abdul-Rahman and S. Hailes. Using recommendations for managing trust in distributed systems. *Proceedings IEEE Malaysia International Conference on Communication*, 1997.

[2] J. Basak, S. Goyal, and R. Kothari. A modified dempster's rule of combination for weighted soruces of evidence. *IBM Research Report*, July 2004.

[3] R. Falcone and C. Castelfranchi. Principles of trust for mas: Cognitive anatomy, social importance and quantification. *Proceedings of the International Conference on Multi-Agent Systems*, 1998.

[4] D. Gambetta. Trust: Making and breaking cooperative relations. *Oxford: Basil Blackwell*, 1998.

[5] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina. The eigentrust algorithm for reputation management in p2p networks. *Proceedings of the Twelfth International World Wide Web Conference*, 2003.

[6] S. Marsh. Formalising trust as a computational concept. *PhD Thesis*, April 1994.

[7] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. *Stanford Digital Library Technologies Project*, 1998.

[8] Faehim Project. http://users.cs.cf.ac.uk/ali.shaikhali/faehim/.

[9] J. Sabater and C.Sierra. Regret: a reputation model for gregarious societies. *Proceedings of the 1st International Joint Conference on Autonomous Agents and Multi-Agents Systems*, 2002.

[10] M. Witkowski, A. Aritikis, and J. Pitt. Experiments in building experiential trust in a society of objective-trust based agents. *Trust in Cyper-societies*, pages 111–132, 2001.

[11] I. H. Witten and E. Frank. Data mining: Practical machine learning tools and techniques with java implementations. *Morgan Kaufmann*, ISBN 1-55860-552-5, 1999.

Ali Shaikh Ali
School of Computer Science
Cardiff University
United Kingdom

Omer F. Rana
School of Computer Science
Cardiff University
United Kingdom
e-mail: o.f.rana@cs.cardiff.ac.uk

Economic Models and Algorithms for Distributed Systems, 25–43
Book Series: Autonomic Systems

Reputation, Pricing and the E-Science Grid

Arun Anandasivam and Dirk Neumann

Abstract. One of the fundamental aspects for an efficient Grid usage is the optimization of resource allocation among the participants. However, this has not yet materialized. Each user is a self-interested participant trying to maximize his utility whereas the utility is not only determined by the fastest completion time, but on the prices as well. Future revenues are influenced by users' reputation. Reputation mechanisms help to build trust between loosely coupled and geographically distributed participants. Providers need an incentive to reduce selfish cancellation of jobs and privilege own jobs. In this chapter we present first an offline scheduling mechanism with a fixed price. Jobs are collected by a broker and scheduled to machines. The goal of the broker is to balance the load and to maximize the revenue in the network. Consumers can submit their jobs according to their preferences, but taking the incentives of the broker into account. This mechanism does not consider reputation. In a second step a reputation-based pricing mechanism for a simple, but fair pricing of resources is analyzed. In e-Science researchers do not appreciate idiosyncratic pricing strategies and policies. Their interest lies in doing research in an efficient manner. Consequently, in our mechanism the price is tightly coupled to the reputation of a site to guarantee fairness of pricing and facilitate price determination. Furthermore, the price is not the only parameter as completion time plays an important role, when deadlines have to be met. We provide a flexible utility and decision model for every participant and analyze the outcome of our reputation-based pricing system via simulation.

1. Introduction

Grid computing is a promising paradigm for sharing IT-resources in large-scale geographically distributed systems through collaboration [14]. It enables to use software and hardware infrastructures from other institutes for an effective sharing of heterogeneous computing resources, data or even high-performance and complex

services [16]. The collaboration in the particle physics Grid Community has been facilitated by virtual organizations (VO) like Atlas or LHCb [8]. Scientists are associated with virtual organizations, which is a loosely-coupled team of people working in the same or closely related projects. They work on the same infrastructure with an interoperable application environment. One of the fundamental aspects of an efficient Grid usage is the optimization of resource allocation among the participants. However, this has not yet materialized. Users tend to send one job several times to the Grid to assure that evaluable results will be returned. The redundant job submission blocks resources, which potentially could be allocated to other, more important jobs. Moreover, site administrators hesitate to always provide all the resources to the VO, in case an internal job is waiting. Then, the instant allocation of the internal job is preferred. This behaviour is comparable to free-riding behaviour in P2P-networks [1]. By introducing prices for jobs the behaviour can be redirected avoiding free-riding. Apparently, we have a conflict in goals. Users are interested in satisfying their resource demand as quickly as possible while the overall goal is to provide a fair and efficient resource sharing. On the supply side providers try to get as much jobs as possible for the highest price they can achieve. Each user is a self-interested participant trying to maximize his utility whereas the utility is not only determined by the fastest completion time, but on the prices as well. Self-interested agents can lead to inefficient market outcome. Enforcing authorities are not always able to detect and punish misbehaviour. Reputation mechanisms help to build trust between loosely coupled and geographically distributed participants [12, 27] to avoid a market of lemons [3]. Future revenues are influenced by users' reputation based on the behaviour of all participants. In Grid networks, the incorrect results returned by a finished job do not reveal information such as, whose fault it was. On the one hand, the provider could have aborted the job. On the other hand, the consumer could have made mistakes in programming the job. Thus reputation mechanisms have to provide tailored metrics to rate provider and consumer. Providers need an incentive to reduce selfish cancellation of jobs, while consumers have to thoroughly analyze their jobs, before they submit them. In this chapter we present a reputation-based pricing mechanism for a simple, but fair pricing of resources. In e-Science researchers do not appreciate idiosyncratic pricing strategies and policies. Their interest lies in doing research in an efficient manner. However, for analytical purposes in the offline mechanism the cost for resource usage are fixed and no price has to be determined, whereas in our more realistic online mechanism the price is tightly coupled to the reputation of a site to guarantee fairness of pricing and facilitate price determination. Furthermore, the price is not the only parameter as completion time plays an important role, when deadlines have to be met. In [12] one of the research questions for reputation mechanisms is how they affect the behaviour of participants in a community. We provide two settings comprising a decision model for every participant and present the optimum of the offline mechanism as

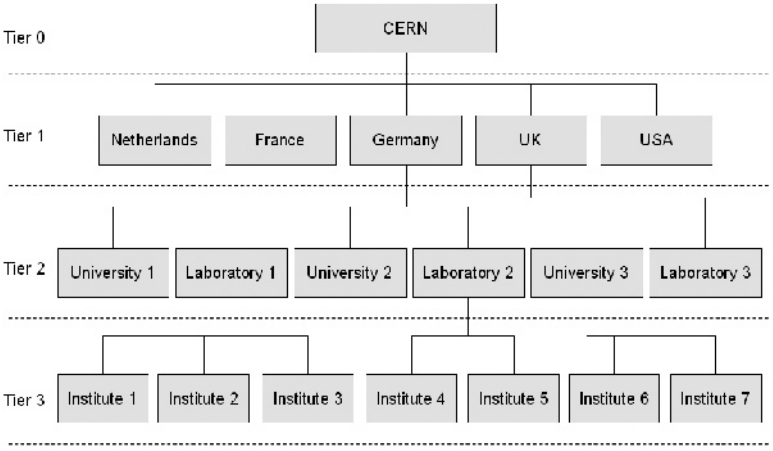

FIGURE 1. Tier levels in the particle physics hierarchy.

well as analyze the outcome of our online reputation-based pricing mechanism via simulation.

2. Offline allocation with fixed price

The community in the particle physics Grid is based on trust between the institutes. Institutes are comprised of several researchers. Researchers typically analyze data and need thus huge amounts of computation power to run simulations and calculations. The management of the resource usage is hierarchically led by CERN (Figure 1). The institutes on Tier 1 as major national institutes are obliged to provide their computing resources for analyzing and saving the data coming from CERN. On this level, a large amount of computing and storage resources are necessary. Tier 2 has a smaller dimension as the institutes get the data from Tier 1 to run simulations. The results of these analyzed data on Tier 1 and Tier 2 have to be shared with all the other participants in the different experiments. Institutes on Tier 2 are larger computing facilities of universities and research laboratories. University departments or a small research group are located on Tier 3. They are not committed to share their resources to the project.

Market based mechanism are promising for offline mechanisms in Grid Computing. Many research papers were published on this topic like [5, 7, 19, 21, 24, 29]. However, this chapter focuses on scientific Grids, where dynamic pricing of resource is undesired. Similar work was done in the distributed network domain. Designing

incentives for participation in peer to peer network was evaluated in [2,11,13,28,33]. All of these papers lack consideration of an allocation mechanism with the goal of fixed pricing vs. load balancing and setting appropriate incentives for the participants in a network.

2.1 Scenario

A Grid network has a set of consumers, called Gridagents $x \in A = \{1, \ldots, a\}$ and a broker, who manages all the resources of the providers. Gridagents can decide on which machine to allocate their jobs. The group of Gridagents submitting a job to the Grid are in $K \subseteq A$. The selected machine by an agent x is defined as m_x and M the number of available machines[1]. They will also have a preferred provider to run their jobs depending on previous good experience with the provider, the customer-friendly Service Level Agreement or the very fast machine. We assume that almost all agents have similar preferences, meaning they favour the same provider most, although they can submit their jobs to less preferred provider (see Figure 2). Every agent has a ranking of the providers and his utility is maximized by the top most provider in his ranking, since he has to pay for every provider the same price (or from the consumer perspective"cost", respectively) c and consequently only the ranking has an influence on the agents' choice. He does not necessarily desire the shortest queue. The ranking is not known by the broker, but he knows that the ranking is a strictly monotonically decreasing function. Moreover, the agents do neither know the current queue length of the providers nor the runtime of their own job. Over time the broker tries to administer the jobs evenly among the available machines by setting the right incentives. He is not allowed to allocate the jobs to another provider without the permission of the Gridagents. Incentives in this mechanism are provided by probabilistically waiving the cost c for the usage at certain providers' machine [22]. Hence, the Gridagent does not have to pay for his job, if he has chosen the right machine. The goal of this broker is on the one side to balance the load over the providers' machine according to the incentives set for the consumers and on the other side to maximize the revenue for the providers. We assume that the broker has an overall utility for the network optimization for calculating the optimal load, which does not necessarily correspond with the Gridagents' preferences.

2.2 Model

The agent's x action N_x is to choose a machine for his job. The selected action is mapped to a probability W_x for each agent x by the function $f : N_1 \times \cdots \times N_a \rightarrow [0,1]^x$. The agent x receives a waiver with a probability $f(m_x)$, if his job is scheduled on machine x. Let $\varphi \in S^n$ be the set of agent strategies and $\varphi(x)$ denotes the distribution over the provider choices for agent x under strategy profile φ. The valuation for a machine m by an agent x is given by $v_x(m)$, which is a strictly

[1] Each provider has exactly one machine, thus the terms are used interchangeably in this chapter.

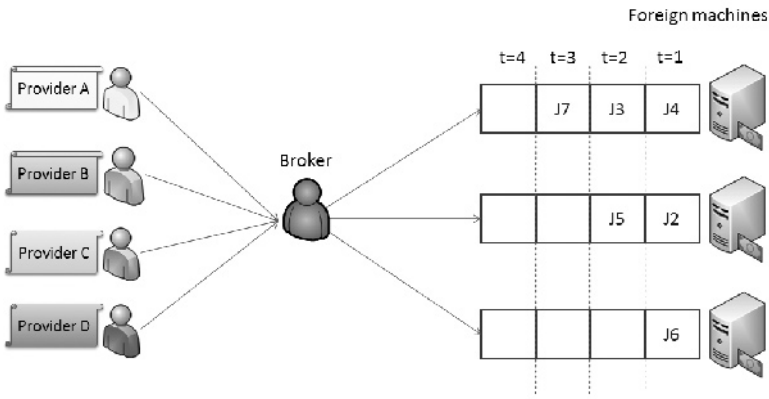

FIGURE 2. Broker allocates requests to providers' free machine.

monotonically decreasing function. Hence, the utility function of a consumer is defined as follows:

$$u_x(\varphi) = \sum_{m_1=1}^{M} \cdots \sum_{m_x=1}^{M} \cdots \sum_{m_a=1}^{M} \Bigg[\Big(\varphi(1)(m_1) \cdot \ldots \cdot \varphi(a)(m_a) \Big) \cdot \Big(v_x(m_x) - \big(1 - f(m_1, \ldots, m_a)_x\big) \cdot c \Big) \Bigg] . \tag{2.1}$$

The idea is to find an equilibrium and force the agents not to deviate from it. The broker has the full information about the providers' queue length. The proposed mechanism is as follows (cf. [22]):

1. Since the broker has full information, he recommends every Gridagent a machine to submit his job to.
2. The Gridagents are allowed to freely choose their provider.
3. The broker decides, on which machines the cost will be waived.

The recommendation of the broker in Step 1 is denoted as $r(x), \forall x \in K$. $\varrho(m)$ defines the number of Gridagents receiving a recommendation to choose machine m. The probability of receiving a waiver $(f(m_x))$ depends on $\varrho(m)$. Step 3 depends on the selection of the machines by the Gridagents. In step 2 they can decide whether to accept or to disregard the advice of the broker, e.g. Gridagents always prefer one machine, they would like to use. Let $h(m)$ be the number of Gridagents selecting the machine m. For simplicity we define the chance for a free machine is same for all agents and thus $f(m) = f(m_x), \forall x \in A$. This leads us to a simpler term than in (2.1)

$$u(m) = v_i(m) - \big(1 - f(m)\big) \cdot c\,. \tag{2.2}$$

Although the Gridagents have the option to select their preferred providers, the mechanism should set incentives for the Gridagents, so that it is rational for them to choose the recommended machine. Let v^l and v^u be a lower and upper bound for all Gridagents' valuation. Consider that a lower bound represents the valuation of an agent for the least preferred machine. Since the Gridagent wants his job run externally on a machine we assume that $v_l > 0$.

In the particle physics scenario Tier 1 resources are dedicated to researchers to do their work. Thus, they have the right to access the resources. Thus, it should be individual rational for the consumers to participate in the mechanism. Subsequently, the lowest valuation for the most preferred machine $v^l(\hat{m})$ among all Gridagents should be higher than cost, $c \leq v^l(\hat{m})$. There is an allocation, where all agents have at least a positive outcome. To motivate the Gridagents for selecting the assigned machine, a Gridagent should evaluate the machine $\check{m}$ with the lowest valuation and his most preferred machine $\hat{m}$ equally:

$$v^l(\check{m}) - \big(1 - f(\check{m})\big) \cdot c = v^u(\hat{m}) - c + \epsilon\,. \tag{2.3}$$

The minimal increment ϵ indicates the strict preference of an Gridagent of the assigned machine. Consequently, the probability of receiving a waiver is

$$f(m) = \begin{cases} \frac{v^u(\hat{m}) - v^l(\check{m}) + \epsilon}{c} & \text{if } \big(h(m) - \varrho(m) \leq 0\big) \wedge \big(\check{s} \neq \hat{s}\big) \\ 0 & \text{otherwise} \end{cases}. \tag{2.4}$$

If $\check{s} \neq \hat{s}$ then the Gridagent has no incentive to deviate from the recommended machine. If a Gridagent deviates from the recommended $r(x) = m$ and selects $\tilde{m}$, while the others follow their recommendation, all the Gridagents, who have chosen $\tilde{m}$, will certainly not receive the waiver. All the other machines will receive the waiver by a probability of $f^b(m), \forall m \in M \setminus \tilde{m}$. Hence, it is a collective punishment for a group of Gridagents, if a single consumer deviates. The definition of the probability in (2.4) gives the Gridagent the incentive to strictly prefer the assigned machine. The Gridagents is always better off, when he submits his jobs to the assigned machine.

3. Reputation-based scheduling and pricing for online allocation

3.1 Scenario

The online setting is more relevant on the Tier 3 level in the particle physics scenario, where each institute has its own resources and these resources are not dedicated to the entire community. Therefore, they share resources with other institutes and have the option to schedule their jobs either on the local machine or to send it to an external site. It depends on the queue length estimation when the job will be finished. Researchers submit their jobs to external sites and anticipate their job will be finished within the expected timeline. Obviously, the jobs of

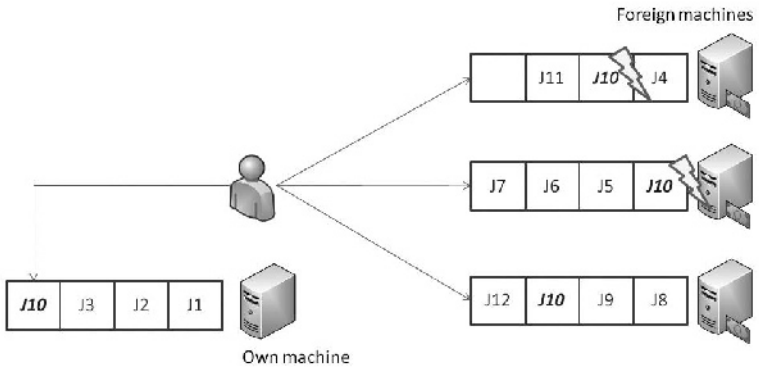

FIGURE 3. Incoming requests for different timeslots.

internal users are always more important than jobs of external users due to selfish manner. Thus, sites tend to cancel current running jobs, if there is an internal job to process, even if the external job is more important or has an earlier deadline. Nowadays, there are no incentives why running jobs of external users should not be cancelled as the institutes do not gain from this job. Users do not have to reciprocate (e.g. payment) to use the Grid resources. Consequently, job requesters will inefficiently consume the offered resource by sending jobs redundantly to the Grid. Figure 3 illustrates the advantage for the consumer sending his job J10 redundantly to several machines. On two of the three foreign machines the job is not executed properly. These jobs are of no value for the consumer. He expects that the other two jobs will deliver valuable results. Without compensation payment he always has the incentive for redundant job submission to avoid the risk of job loss. However, J10 delays other jobs (e.g. J11), which could be more important.

The goal of the system is to achieve a fair allocation of resources and enforce obedient behaviour of the participating agents. Generating a reliable Grid platform in the e-Science community for distributed resource sharing and collaboration requires mechanisms to solicit and predict resource contributions of individual users [9]. Buragohain suggested an incentive mechanism for P2P file sharing, where users with higher service provision have a higher probability to be accepted by others for downloading files. Every user has costs for offering files and he can gain from offered files by other users. The approach in this chapter is to differentiate between the services a user provides. A user with a high contribution is more likely to be accepted than a user with a low contribution. This analysis framework can be used to identify the benefit of participating in the network after the transaction. Burgahoin's game theoretic analysis framework is not applicable for Grid Computing, since it does only focus on resources that are reusable within one timeslot. In P2P networks a file can be downloaded by several peers at the same time (parallel resource usage). CPU cycles, however, cannot be shared within

one timeslot without loss of quality. Subsequently, the assignment of resources does not only depend on his provision level, but also on the available resources at the requested timeslot. Moreover, we do not reject requests based on probability. Jobs which have been submitted to a site must be accepted. The introduction of payment can solve the problem of inefficient resource usage. Every user has to pay a certain amount of money to receive resources. The amount of money has to be limited, since real money in scientific Grid networks is undesirable. Instead virtual currencies or credits can be implemented like Karma or Nuglets [10, 31]. This induces further problems as virtual credits are used to price resources. Users or site administrators have to decide how to determine the price of a resource depending on capacity, demand and availability of resources. This entire decision process requires time from researcher, which distracts them from their research work. Another (simpler) option is to have a fixed price for resources, e.g. 1 credit per CPU/minute. The prices need not be determined dynamically and an incentive is provided to stop over-consumption of resources. However, fixed price fails to set incentives to behave compliantly as site administrator can cancel job at any time. They will accept a short-term loss in payment made by the current running job. In this case the own job, which has to be finished before the deadline and has a higher valuation than the fixed payment, will replace the running job. In this chapter a fixed-price scheme is extended by a reputation mechanism to enhance incentives for collaboration in the Grid network. Prices usually reflect the supply and demand. The proposed pricing is advantageous as it reflects the service level. An automatic adaptation of the price according to users' behaviour allows setting the right incentives for collaborative work resulting in an improved exchange of resources compared to the current real-world solution. Furthermore, a decision framework for cancelling a job is presented to depict the scenario in a scientific Grid network.

3.2 Model

The main idea has been derived from [17], where the authors propose a reputation-based pricing for services in P2P networks based on the provided quality of service. Deadlines and completion time are not considered in their utility function and thus not suited for Grid. Our utility function comprises these parameters. We adapted the online scheduling mechanisms from [26] and [15]. Porter's utility function is not based on the length of the job, but on the valuation for the job. The mechanism contemplates when and how a job has to be submitted and users can report true or false values for job length or job valuation. We assume that a long job has a bigger impact and a higher risk to be cancelled than a short job and the values of a job are reported truthfully. Typically, long jobs comprise high effort in programming and thus the impact of the results is high. Due to their long running time the probability increases that the job will be cancelled. Moreover, the formal analysis of Porter is based on mechanism design for a single machine, whereas in our case we consider m machines in a simulation. Heydenreich et al. [15] propose

a mechanism called Decentralized LocalGreedy Algorithm (DLGM). There is no central planner to allocate jobs to the different nodes. Instead, jobs ask for the completion time and payment on each machine and decide on which machine they want to be scheduled. Jobs can report their value and get a higher priority and be executed earlier than previously allocated jobs. Deadlines of the jobs are not taken into account. Furthermore, the option that a user (or the machine owner) can cancel the current running job was not analyzed. It makes a new option for decision available. In e-Science Grid users are researchers who are sending jobs to the Grid and consume subsequently from other Grid research institutes. In our model we will consider sites as consumer and provider. Other papers (e.g. [20]) distinguish between provider and consumer as two different persons/institutes, whereas in our case the decision model is dependent on both roles (cf. [9]). Thus, sending and receiving jobs has an impact on the decision for a site in both roles. To distinguish between the provider and consumer role we will name the consumer as jobs and provider as machines. Jobs and machines can belong to the same user. For simplicity, this model implies without loss of generality that one site has only one user, where every user has a reputation. The calculation of the reputation value is not fixed to a certain scheme. Promising examples for reputation mechanisms are [4] for Grid networks as well as [32] and [18] known from P2P networks. Similar to the mentioned examples we assume that users in Grid report the feedback truthfully and act rationally.

3.3 Parameter

Preferences of a user are expressed by the utility function. It is crucial for defining the relation between the loss and the benefit of a job on a machine at a certain time. Besides obvious and essential characteristics of a utility function further requirements have to be met:

- A job which is completed after the deadline has a value of zero. It does not have any value for the user, if the job is finished too late.
- The risk of a job cancellation increases, if the provider has a lower reputation.
- The cancellation of a job must have a direct impact on the future income.

We again consider a scenario with a set of Gridagents $A = \{1, \ldots, a\}$, who participate in the network by providing resources and submitting jobs. Resources are homogeneous. Every agent $x \in A$ can send one or more jobs j_x to the Grid in one timeslot $T = \{1, \ldots, t\}$. The jobs are defined by a processing time $p_j > 0$ (runtime of job), and a deadline $d_j > 0$ (when the job should be finished). The incoming jobs are always able to meet the deadline, if they are instantly started. Every job requests the machines in the network ($M = \{1, \ldots m\}$ with $x = m$) for their queue time $q_{jm}(t)$. This approach is comparable to the DLGM setting [15]. The completion time $C_j(m)$ is defined as $C_j(m) = p_{j_x m} + q_{jm}(t)$, where $p_{j_x m}$ denotes the remaining time for the current running job j_x from agent x on machine m. Every machine has a reputation value $r_m \in [0, 1]$. Porter proposed a

utility function based on a hard deadline. Every user has an expectation about the latest finishing date of a job. The job is worthless, if it is finished after the deadline and it is thus cancelled (Porter 2004). We do not abort jobs, if they are waiting in the queue and probably will not match the deadline. We assume that jobs can still match the deadline, if preceding jobs are cancelled and replaced by shorter jobs. Then, the completion time will be reduced. The option of cancellation is only considered, if a new job cannot be finished before the deadline. The valuation for the job's laxity is determined by the parameter V_j. Thus, the utility is defined as

$$U_{jm}(t) = \mu * \left(D_j \geq \frac{1}{\sqrt{r_m(t)}} * C_{jm}(t) + t \right) * V_j - \pi_{jm}(t)\,. \tag{3.1}$$

We use the notation according to Porter, where $\mu(\cdot)$ is an indicator function, which returns 1, if the argument is true, and zero otherwise (requirement 1). The deadline should be bigger than the sum of the completion time and the current timeslot. Furthermore, every machine is evaluated by the risk of job cancellation. We introduce a risk factor $\frac{1}{\sqrt{r_m(t)}}$ to fulfil requirement 2. If the machine has a high reputation, the job will more likely be finished before deadline. $\pi_{jm}(t)$ is the total payment job j pays to the machine m. We propose a reputation-based pricing, which enables a direct price determination based on the reputation of the provider. A provider with a higher reputation will consequently receive a higher income per timeslot. If the reputation decreases, the price will decrease, too. Let the price per timeslot be $v_{jm}(t) : [r^{min}, r^{max}] \rightarrow \mathbb{R}$ (requirement 3). In the remainder of this chapter it is simplified to $v_{jm}(t) = rm(t) \in [0, 1]$. The total payment of a user to the machine m is $\pi_{jm}(t) = p_j * v_m(t)$. A consumer can rate the provider, if the job was cancelled, finished too late or successfully. The provider is not rated negative, if the deadline was matched, although the promised finishing date was delayed. At the beginning of the allocation the price is set accordingly to the reputation. The utility of a job can be positive or negative, since the payment can be higher or lower than the valuation of a job. On the contrary, the definition of DLGM only allows negative utility.

3.4 Sellers' and buyers' action space

When a job is created, the agent has the action space S for the job with $S = \{$*run job on own machine, run job on foreign machine, cancel running job of other agent*$\}$. Usually, the third option is the best, if no reputation and prices are considered, since the agent is not punished for his misbehaviour. We only consider the option to cancel the running job. The replacement of a job in the queue is not taken into account in this setting.

The decision process is as follows. At first, the agent calculates the utility, if his job is scheduled on his machine. Although the agent does not have to pay himself ($\pi_{jm}(t) = 0$), the agent has opportunity costs, since no other foreign jobs can run on this machine and the income is missing for this period of time. In the

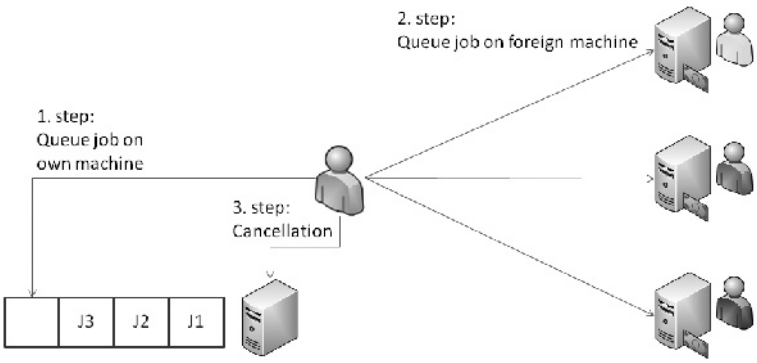

FIGURE 4. Decision process of the user.

next step he analyses whether a better utility can be gained by running the job on a foreign machine. The completion time has to be lower, because the price decreases the utility ($\pi_{jm}(t) \geq 0$) compared to scheduling on his machine. As third option users have the ability to cancel running jobs on their own machine while processing their job instantly (Figure 4). This is a big advantage, when queues are very long due to high demand and the completion time of a job extends the deadline on all machines, e.g. it does not get finished within time. We assume that a user would never cancel his own job on his machine. From the consumer perspective it is only attractive for the consumer to schedule his job on another machine, once the current running job is not from another provider. Otherwise, it is reasonable to cancel the running job, because he will obtain the lowest completion time (without reputation and payment). On the one hand, payments decrease the benefit of scheduling the job on another machine, since for their local site the user is not required to pay. Consequently, it is less attractive to send jobs to others. On the other hand, the cancellation of jobs results in a negative outcome for the machine owner, because he will not receive any payments and he will be punished by a lower reputation. Queues on other machines may comprise fewer jobs and thus attract users to schedule their jobs on other machines.

By missing the deadline the results of a job create no value. Henceforth, the provider faces two effects: payment and reputation loss. Payment loss arises by replacing the current running job by a new job and delaying other jobs in the queue. Delay can result in missing the deadline. Finished jobs beyond deadline are not being paid. The loss is calculated by summing up the excepted payment for all delayed jobs including the cancelled job:

$$l_m^{delayLoss}(t) = \sum_{k=1}^{Q} \pi_{km}(t) * \mu\big(D_j < \hat{C}_{jm}(t) + t\big)\,. \tag{3.2}$$

Delayed jobs and the cancelled job have the opportunity to rate the provider. Apparently, they will submit a negative rating and the provider will face a reputation loss. The number of negative ratings is $R_m^{neg}(t) = \sum_{k=1}^{Q} \mu(D_j < \hat{C}_{jm}(t) + t)$. Depending on the reputation mechanism the negative ratings will lower the reputation of agent a possessing machine m. Then, the agent has to collect $R_m^{pos}(t)$ positive ratings to regain his former reputation. $R_m^{pos}(t)$ is the number of required jobs, which rate the machine positively. $R_m^{neg}(t)$ and $R_m^{pos}(t)$ need not be equal, i.e. in asymmetric reputation mechanisms, where it is more difficult to receive a good reputation than a bad reputation. Next, it has to be analyzed how long it will take to receive the required jobs. As there are jobs already in the queue meeting the deadline the number of required jobs for obtaining the old reputation value is $\hat{J}_m(t) = R_m^{pos}(t) - \sum_{k=1}^{Q} \mu(D_j < \hat{C}_j m(t) + t)$. Since the machine will have a lower income due to the reputation loss and thus a lower price, the compensation is based on the current queued jobs and the incoming rate $\hat{\lambda}$ of jobs on the machine m in the future timeslot. Let the prospective jobs arrive according to a predefined distribution and have a processing time equalling the mean $\bar{p}$. The incoming rate of jobs is derived from the history. We assume that jobs will arrive according to former income rate. We use the exponential smoothing method to predict the jobs arriving in the future. Subsequently, the number of expected jobs arriving in timeslot $t+1$ is $J_m(t+1) = \alpha * y_t + (1-\alpha) * J_m(t)$. The required number of timeslots to restore the old reputation encompasses the duration of all jobs in the queue and the expected runtime of future jobs:

$$t_m^{required} = \bar{p} * \hat{J}_m(t) + \sum_{k=1}^{Q} p_k \,. \tag{3.3}$$

The jobs, which arrive before the reputation is restored, create a loss for the provider, since they have to pay a lower price. The reputation value including $R_m^{neg}(t)$ ratings $r_m^{neg}(t)$ and the current value is $r_m^{pos}(t)$. We average the price the incoming jobs have to pay until $t_m^{required}$ by $v_j m(t) = (r_m^{pos}(t) - r_m^{neg}(t))/2$. Knowing the number of expected jobs and the excepted payment, the expected loss can be determined, if the agent cancels the running job: $l_{jm}^{repLoss}(t) = t_m^{required} * v_{jm}(t)$. The total utility for cancelation is

$$U_{jm}^{cancel}(t) = \mu * \big(D_j \geq p_j(t) + t\big) * V_j - l_m^{delayLoss}(t) - l_{jm}^{repLoss}(t) \,. \tag{3.4}$$

Users can decide whether the utility gained by the cancellation exceeds the utility of a regularly scheduled job. In our simulation we only consider this option, when the job will fail the deadline due to large queues.

3.5 Reputation mechanism

The selection of a reputation mechanism is crucial for the mechanism. Different reputation mechanism will influence the user's decision function. The authors of [23] provide a taxonomy for identifying properties of reputation mechanisms

to choose the right mechanism for the right setting. Although the taxonomy was mainly developed for P2P networks, it is applicable for Grid networks as well. They distinguish between information gathering, scoring and ranking and response. The first category comprises the precondition to create identities, the information sources and the level of information detail. Scoring and ranking defines the input data and the output data of a reputation mechanism. Response is the action a user can take or the action space a user is limited to, i.e. users who have a low contribution level also have a low download capacity in P2P networks. In the particle physics Grid we deal with registered identities, which have to be certified by a certification authority. This process is only for authorising the user to participate in the network. The site administrator is unaware of whom the current running job belongs to. Jobs are mapped by a proxy identifier, which is managed by the resource broker. Thus, the circumvention of the reputation mechanism like whitewashing and Sybil attacks [11] are impossible in Grid networks, since every certificate application is thoroughly analyzed by several authorities. However, the consumer is unknown to the provider due to anonymity. Information sources can either be local reputation or global reputation. In this chapter we restrict our attention to global reputation, since we have centralized authority to gather and disseminate this information and we assume to have agents rating honestly [23,30]. The level of information detail will be reduced to aggregated information about former behaviour to keep it as simple as possible. The reputation will be represented by a single value. Strategic behaviour based on the former actions is not taken into account. The input data for computing the reputation value weigh current ratings higher than old ones. Jurca and Faltings preferred an even simpler mechanism by deriving the reputation value as follows: fracnumber of good ratingsnumber of total ratings. We apply this simple reputation mechanism as in [17,30] and adapt it with a straightforward decay function [4,6]. The rationale behind this is that older reputations are less important [34]. The quasi-decay function takes only the last g ratings and weights them equally. This approach induces that the order of the submitted rating is essential. Thus, ratings, which came in first, will be excluded first after g ratings.

4. Simulation and implementation

In this section, we present the first results from our simulation. Our results show that reputation-based pricing gains higher exchange of resources and less cancellation of jobs than fixed price schemes.

4.1 Setting

Currently the particle physics researchers do not have to pay for the usage of Grid resources. We therefore set up a scenario without payment and a scenario including reputation based pricing. The agents are fully trusted at the beginning. They start

Table 1. Simulation results.

	Number of cancelled jobs	Deadline not matched		Finished foreign scheduled jobs	Utility
		Own jobs	Foreign jobs		
Rep. pricing	35	6	22	98.1%	103,42%
Fixprice = 0	301	33	74	94.6%	100,00%
Fixprice = 0.5	309	28	64	93.3%	100,16%
Fixprice = 1	328	22	55	93.1%	100,32%

with a reputation value $r_m(0) = 1$. The simulation is round-based and comprised 1000 rounds in each 20 runs. Every round jobs were created according to a Poisson distribution with $\lambda = 0.25$ for each agent. 50 agents were providing their resources and interacting with each other. Jobs had a completion time and stayed on a machine until the job was done or cancelled. The job processing time was derived from a truncated normal distribution with mean = 3 and deviation = 2. Only positive durations were allowed. Every job had a deadline which was uniformly distributed between 10 and 40 timeslots. The valuation of a job for the job owner was uniformly distributed between 1 and 10. A job was of no value, if it was finished after the deadline. To benchmark the reputation-based pricing we used a fixed price scheme with $p = 0$, $p = 1.0$ to demonstrate the effect of prices on the scheduling outcome. A fixed price with p = 0 represents the current Grid where no payment is necessary. The highest payment in the reputation-based scenario is $r^{max} = 1$.

4.2 Results

The goal of the proposed mechanism is to show the effect of reputation-based pricing and the benefit for the particle physics Grid. One metric is to view the amount of cancelled jobs. The less jobs are cancelled, the higher the trust in the network. Table 1 illustrates the results. The reputation-based pricing enforces site administrator not to cancel jobs, since their behaviour is documented by the reputation. For the fixed price scenarios there is negligible discrepancy between each setting. Overall, there were about 900 attempts to cancel a job for fixed prices and about 750 for the reputation-based pricing. It encompasses the attempts to cancel the own jobs as well, which was excluded in our setting. Consequently, one third of the jobs were cancelled in the fixed price setting and only 5% in the reputation-based pricing. This is a significant reduction and a strong enhancement of trust in the Grid network. Another metric considers the number of jobs, which have not met their deadline. Grid users rely on the jobs they sent to other sites. When jobs regularly do not meet the deadline due to cancelation, users will distrust other sites, because participation in the Grid will not be individual rational [29]. Accordingly, jobs waiting in the queue may not meet their scheduled deadlines as in the meantime the machine has cancelled the running job in favour of a longer

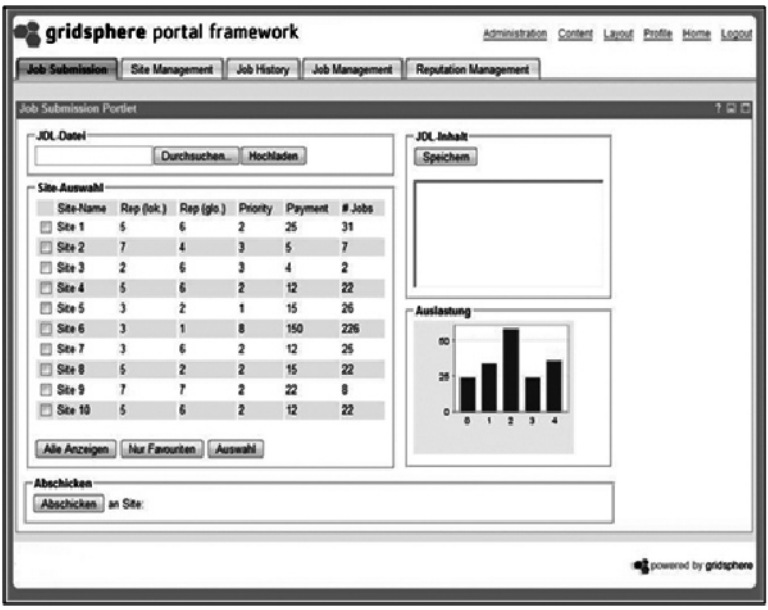

FIGURE 5. Billing the grid portal.

own job. As shown in table 1 there is a discrepancy between the reputation-based pricing and fixed pricing. The jobs of a user running on his machine were always lower than foreign jobs for all four schemes. Job cancellation affected queued jobs including own jobs. The loss of own jobs by failing the deadline was taken into account. Therefore, users avoided to cancel jobs, when cancellation had an impact on too many own jobs in the queue. Looking at all foreign scheduled jobs 98.1% of the jobs were finished successfully with the reputation-based pricing scheme. The fixed price schemes are above 93%. The discrepancy results from previous two metrics. The overall utility gained from the reputation model was 3% higher than the fixed price scheme. A better utility could be achieved by enhancing the agents with more intelligent tactics. The selection of sites can be differentiated according to value of a job, reputation, deadline and payment in a more detailed approach. Weighing these four parameters can attain a better outcome for all agents.

4.3 Application

The reputation-based pricing mechanism is currently implemented in a billing infrastructure called Billing the Grid (BtG). The goal of this infrastructure is to provide a reputation and billing mechanism for particle physics scientists. The introduction of incentives in the scientific Grid will enable an efficient utilization of existing resources. Consumers have the incentive to avoid the submission of redundant jobs as they have to pay for each job. Site administrators have the incentive

to keep jobs running and avoid system downtime, since it results in payment loss. This infrastructure provides a graphical user interfaces based on the Gridsphere framework [25]. The portlets allow users to submit jobs, get detailed information about the sites and to rate users according to their behaviour (Figure 5). The portlets send jobs to and receive information from the middleware gLite[2]. Users describe their job requirements in a JDL-file (job description language). The resource broker matches the requirements with the available resources and sends the job to the aappropriate site. Consumers still have the ability to choose a certain site for their job. BtG supports the user's decision by providing additional information about the sites' reputation. Beside the reputation-based payment model and the global reputation, users can maintain a local reputation table. In case, a specific type of job is not able to run on a particular machine, they can rate the site with a low reputation. Next time, this machine can be avoided, although it has a good global reputation. The graphical user interface eases the management of finished jobs, favourable sites and credit account as well as rating of jobs.

5. Conclusion

Grid computing facilitates IT resource sharing among distributed organizations. Particle physics Grids are currently running inefficiently due to redundant job submissions. At first, we analyzed an offline setting, which considers reputation in an allocation mechanism to achieve a fair distribution of resource usage. The offline mechanism is especially interesting for public resources. We further proposed an online mechanism considering a reputation-based pricing approach. This pricing scheme has the advantage of setting incentives for providing resources and consuming them prudently. But, the effort for determining the right price at the right time is avoided, since scientists prefer rather to concentrate on their own research work than make economic decisions. In fact, the dynamic of prices are based on the reputation of the users, which enforce them to behave cooperatively and elude the price specification. We presented a decision model to denote the behaviour in scientific Grids and showed in a simulation that reputation-based pricing can improve collaborative interaction. These results strengthen our approach and illustrated future avenues. A deeper analysis of agent-based modelling and decision will allow a more realistic simulation. Different strategies can be tested within the decision model. Moreover, appropriate reputation mechanisms can be analyzed regarding attacks and incentives in this decision framework.

Acknowledgements

This work was supported by a grant from the Ministry of Science, Research and the Arts of Baden-Wuerttemberg (AZ: 23-7532.24-38-14/1) to the authors.

[2]http://www.glite.org

References

[1] E. Adar and B. A. Huberman. Free riding on gnutella. *First Monday*, September 2000.

[2] M. Adler, R. Kumar, K. Ross, D. Rubenstein, T. Suel and D. D. Yao. Optimal peer selection for p2p downloading and streaming. In *INFOCOM* 2005*, 24th Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 1538–1549, 2005.

[3] G. A. Akerlof. The market for lemons: Quality uncertainty and the market mechanism. *The Quarterly Journal of Economics*, 84(3):488–500, 1970.

[4] B. K. Alunkal, I. Veljkovic, G. von Laszewski and K. Amin. Reputation-based grid resource selection, 2004.

[5] A. Auyoung, B. Chun, A. Snoeren and A. Vahdat. Resource allocation in federated distributed computing infrastructures. In *Proceedings of the* 1*st Workshop on Operating System and Architectural Support for the On-demand IT InfraStructure*, 2004.

[6] F. Azzedin and M. Maheswaran. Evolving and managing trust in grid computing systems. In *IEEE Canadian Conference on Electrical & Computer Engineering* (*CCECE*'02), pages 1424–1429, 2002.

[7] R. Bapna, S. Das, R. Garfinkel and J. Stallaert. A Market Design for Grid Computing. *INFORMS Journal on Computing*, 20(1):100–111, 2008.

[8] R. Berlich, M. Kunze and K. Schwarz. Grid computing in europe: from research to deployment. In *ACSW Frontiers*'05*: Proceedings of the* 2005 *Australasian workshop on Grid computing and e-research*, pages 21–27, Darlinghurst, Australia, Australia, 2005. Australian Computer Society, Inc.

[9] C. Buragohain, D. Agrawal and S. Suri. A game theoretic framework for incentives in p2p systems. In *P2P*'03*: Proceedings of the* 3*rd International Conference on Peer-to-Peer Computing*, Washington, DC, USA, 2003. IEEE Computer Society.

[10] L. Buttyan and J. P. Hubaux. Stimulating cooperation in self-organizing mobile ad hoc networks. *ACM/Kluwer Mobile Networks and Applications*, 8(5):579–592, 2003.

[11] A. Cheng and E. Friedman. Sybilproof reputation mechanisms. In *P2PECON*'05*: Proceeding of the* 2005 *ACM SIGCOMM workshop on Economics of peer-to-peer systems*, pages 128–132, New York, NY, USA, 2005. ACM Press.

[12] C. Dellarocas. The digitization of word-of-mouth: Promise and challenges of online feedback mechanisms. *Management Science*, 49(10):1407–1424, 2003.

[13] M. Feldman, K. Lai, I. Stoica and J. Chuang. Robust incentive techniques for peer-to-peer networks. In *EC*'04*: Proceedings of the* 5*th ACM conference on Electronic commerce*, pages 102–111, New York, NY, USA, 2004. ACM Press.

[14] I. Foster, C. Kesselman and S. Tuecke. The anatomy of the grid: Enabling scalable virtual organizations. In *Euro-Par*'01*: Proceedings of the* 7*th International Euro-Par Conference Manchester on Parallel Processing*, pages 1–4, 2001.

[15] B. Heydenreich, R. Müller and M. Uetz. Decentralization and mechanism design for online machine scheduling. In *Proceedings of the Scandinavian Workshop on Algorithm Theory* (*SWAT*), pages 136–147. Springer, 2006.

[16] J. Joseph, M. Ernest and C. Fellenstein. Evolution of grid computing architecture and grid adoption models. *IBM Systems Journal*, 43(4):624–645, 2004.

[17] R. Jurca and B. Faltings. Reputation-based pricing of p2p services. In *P2PECON'05: Proceeding of the* 2005 *ACM SIGCOMM workshop on Economics of peer-to-peer systems*, pages 144–149, New York, NY, USA, 2005. ACM Press.

[18] S. Kamvar, M. Schlosser and H. Garcia-Molina. The eigentrust algorithm for reputation management in p2p networks. *Proceedings of the Twelfth International World Wide Web Conference* (*WWW*), pages 640–651, 2003.

[19] J. Kay and P. Lauder. A fair share scheduler. *Communications of the ACM*, 31(1):44–55, 1988.

[20] Y. K. Kwok, S. Song and K. Hwang. Selfish grid computing: Game-theoretic modeling and nas performance results. *IEEE International Symposium on Cluster Computing and the Grid*, pages 1143–1150, 2005.

[21] K. Lai, L. Rasmusson, E. Adar, L. Zhang and B. A. Huberman. Tycoon: An implementation of a distributed, market-based resource allocation system. *Multiagent and Grid Systems*, 1(3):169–182, 2005.

[22] K. Leyton-Brown, R. Porter, B. Prabhakar, Y. Shoham and S. Venkataraman. Incentive mechanisms for smoothing out a focused demand for network resources. *Computer Communications*, 26(3):237–250, February 2003.

[23] S. Marti and H. Garcia-Molina. Taxonomy of trust: Categorizing p2p reputation systems. *Computer Networks*, 50(4):472–484, March 2006.

[24] D. Neumann and J. Stößer. Greedex – a scalable clearing mechanism for utility computing. *Electronic Commerce Research*, 8(4):235–253, 2008.

[25] J. Novotny, M. Russell and O. Wehrens. Gridsphere: an advanced portal framework. In *Proceedings of the Euromicro Conference*, pages 412–419, 2004.

[26] R. Porter. Mechanism design for online real-time scheduling. In *Proceedings of the ACM Conference on Electronic Commerce* (*EC'04*), pages 61–70, 2004.

[27] P. Resnick, B. Zeckhauser, E. Friedman and K. Kuwabara. Reputation systems. *Communications of the ACM*, 12(43):45–48, 2000.

[28] S. Sanghavi and B. Hajek. A new mechanism for the free-rider problem. In *P2PECON'05: Proceedings of the* 2005 *ACM SIGCOMM workshop on Economics of peer-to-peer systems*, pages 122–127, New York, NY, USA, 2005. ACM.

[29] B. Schnizler, D. Neumann, D. Veit and C. Weinhardt. Trading grid services – a multi-attribute combinatorial approach. *European Journal of Operational Research*, 187(3):943–961, 2008.

[30] J. D. Sonnek and J. B. Weissman. A quantitative comparison of reputation systems in the grid. In *The 6th IEEE/ACM International Workshop on Grid Computing*, pages 242–249, 2005.

[31] V. Vishnumurthy, S. Chandrakumar and E. Sirer. Karma: A secure economic framework for peer-to-peer resource sharing. In *Workshop on the Economics of Peer-to-Peer Systems*, 2003.

[32] L. Xiong and L. Liu. Peertrust: Supporting reputation-based trust for peer-to-peer electronic communities. *IEEE Transactions on Knowledge and Data Engineering*, 16(7):843–857, July 2004.

[33] B. Yu and M. P. Singh. Incentive mechanisms for peer-to-peer systems. In *Second International Workshop on Agents and Peer-to-Peer Computing*, pages 77–88. Springer, 2003.

[34] G. Zacharia, A. Moukas and P. Maes. Collaborative reputation mechanisms in electronic marketplaces. In *Proceedings of the* 32*nd Annual Hawaii International Conference on System Sciences*, pages 7–14, 1999.

Arun Anandasivam
University of Karlsruhe (TH)
Englerstraße 14
76131 Karlsruhe
Germany
e-mail: `anandasivam@kit.edu`

Dirk Neumann
University of Freiburg
Platz der Alten Synagoge
79085 Freiburg
Germany
e-mail: `dirk.neumann@is.uni-freiburg.de`

Economic Models and Algorithms for Distributed Systems, 45–61
Book Series: Autonomic Systems

Trust-oriented Utility-based Community Structure in Multiagent Systems

Georgia Kastidou and Robin Cohen

Abstract. The problem we address in this chapter is how to design the community structure of a multiagent system in such a way that agents join the communities that will maximize their utility and communities accept the agents that will maximize their utility, towards a stable and productive multiagent system. In order to accomplish this goal, we propose allowing communities to exchange information about the reputability of agents. In particular, if agent a_1 exists in community c_1 and would now like to join c_2, c_2 will ask c_1 for the reputation rating of a_1 and then decide whether to allow the agent to join. Allowing for the sharing of reputation ratings then requires i) a method for determining the truthfulness of the reputation reports ii) an incentive mechanism to encourage the sharing of information iii) some consideration of privacy of information within the system. In order for agents to make effective selection of communities in which to participate, it is also ideal for these agents to learn more about the utility that current agents in the community enjoy and about the tendency for the community to be truthful, when it reports reputation ratings of agents. We present a reputation sharing system that promotes effective community structure, along with examples to demonstrate the benefit of this particular approach.

Keywords. Grid computing, reputation, pricing, incentives

1. Introduction

Within the field of artificial intelligence, researchers have studied the development of distributed solutions to problem solving on behalf of users. Multiagent systems [17] are collections of intelligent programs known as software agents, acting on behalf of their human users. This approach has been used for instance to facilitate the design of a P2P system, with the agents helping to model other users in the community. Designing frameworks to assess the trustworthiness of agents in multiagent systems is a topic that has received much attention from researchers, in

recent years. These models vary from ones that rely on direct experience (e.g. [1]), where agents can learn over time which other agents to trust, based on past experiences and ones that make use of a social network (e.g. [2, 3]), where agents ask others in their community about the reputability of agents with whom they have had relatively little experience. When social networks are used, agents need to then assess the trustworthiness of those agents that provide information, as well. Often the setting for this research on trust modeling is that of electronic marketplaces, where buying agents select selling agents, based in part on ratings (about the sellers' trustworthiness in delivering goods) provided by advisor agents. The problem we are interested in addressing is how to now have communities reasoning about the trustworthiness of their agents and also have communities sharing reputation ratings of agents between them. In this setting, an agent would accumulate a reputation rating (typically a value in the range [0,1] where 0 is the least trustworthy and 1 is the most) in the community in which it currently participates, but that rating may be carried to other communities which it is interested in joining. In addition, we are interested in trust models that determine the value of an agent based not only on the quality of its contributions but also on the extent to which it has become an active participant in the community. This is relevant, for instance, in file sharing communities where agents are expected to upload files with some frequency (but also to be providing files that the rest of the community will actually enjoy). In allowing communities to share reputation ratings, the aim is for untrustworthy agents to be detected more efficiently, resulting in an overall system that has the following benefits:

- trustworthy agents are identified, so that they can enjoy benefits right away.
- the average contribution of agents in the community should increase, since agents will be motivated to contribute; otherwise, this will have a negative impact on their reputation
- the services and information provided will be more useful, since they will better match the profile of each community.

Our research constitutes an effort to develop new incentive mechanisms for distributed systems in collaborative environments. Our focus is on supporting the modeling of trustworthiness of agents of use in community-based environments, such as P2P file sharing and on reducing the good behaviour of participants. As such, the ultimate aim of this research is to provide the basis for the kinds of commercial systems that are being built to enable effective online file sharing

2. The approach

In order to address the problem presented in Section 1, we require a methodology that:

- models trustworthiness and reputation of agents, within communities.
- provides incentives for communities to share reputation ratings of agents.
- allows agents to reason about which communities it is beneficial to join.

In addition, the overall aim is to develop an approach that can maximize the utility of the agents and of the communities. In this section, we outline our proposed model and illustrate it with examples. We then include a brief discussion of how this model can be further validated, towards delivering high utility to both agents and communities.

2.1 Communities reasoning about agents

In a multiagent system there are two ways to motivate agents to contribute and to be truthful. The first one is based on the assumption that the agents can see their long term benefits. More specifically, the agents can reason that if all of the agents in the system are truthful and contribute then eventually they will be able to utilize better services.

As agents represent users and thus act based on the users' strategies, the assumption that all the agents would be able to see the long term benefits they might acquire can be seen as unrealistic.

In order to deal with this problem, the use of incentives can be introduced. From our perspective, the idea of using incentives is to 'motivate' agents to be truthful by 'threatening' to directly or indirectly limit their access to the system privileges.

For our particular situation we assume that there may indeed be an incentive mechanism to encourage truthful behaviour of agents in their communities (e.g. in P2P file sharing communities, restricting the bandwidth for unreliable agents). But we have a new element as well to promote good behaviour from agents: their reputation in current communities may be shared with the new communities they are interested in joining.

In this framework, each community must reason about which agents to accept, based on a modeling of their trustworthiness. Part of this reasoning is an interpretation of reputation ratings of these agents provided by other communities. We present some insights into how these processes can be modeled and then return to provide a general overview of the community reasoning process.

Included in our discussion is a method to motivate communities to freely share accurate reputation ratings of their agents. For this, we present two incentive-based approaches. The first is a very simple approach where each community has a utility function that is increased each time the community provides information to another community, and is decreased each time a community acquires information regarding an agent or each time a community is caught to be lying about an agent. In our effort to carefully complete the design of this approach we came across several problems that led us to consider a second approach. The second approach is an auction-based approach which retains certain elements of the first approach but exploits valuable properties of auction-based mechanisms to address certain challenges with the first approach.

2.1.1 Modeling the trustworthiness of agents

Several multiagent systems researchers have developed systems for modeling the trustworthiness of agents. These include direct experience models such as that of Tran [1] that use reinforcement learning to adjust, over time, the reputation rating of an agent, using thresholds to determine whether an agent should be placed in a reputable, disreputable or neither reputable nor disreputable set. When an agent needs to be selected, a suitable agent is first sought from the trustworthy set and disreputable agents are ignored. There are also approaches that make use of a social network of agents providing third party reports, where the trustworthiness of these reports must then be judged as well before the overall reputability of the agent can be assessed, in a kind of aggregation (e.g. [2,3]) and where untrustworthy reporters can be ignored or discounted.

In order to model the local reputation of an agent, we have identified several features that we feel are important to retain, drawn from aspects that are considered by various existing trust and reputation models:

1. the ratings an agent received for the services it provided to other agents, sensitive to their rating habits (e.g. are these agents in general lenient or strict etc).
2. how often the agent was rated.
3. the importance of each service that an agent provides (e.g. the value of the service, the novelty of the service, the degree to which the service benefits the whole community).

In order to judge both the level of participation and the quality of the contributions from agents in the community, we propose the following:

Quality of contribution is assessed by other agents in the community, each time the agent engages in a transaction with another agent. This is reported to a central entity, where the trustworthiness of the agent and subjectivity of the reporter is evaluated and the overall reputation of the agent is updated, taking into consideration the suitably weighted rating value provided by this agent. We refer to this aspect of the agent as its Internal Trustworthiness (what each agent reports) and its local reputation (the aggregation of internal trustworthiness values in the system).

Level of participation of the agent can be measured by tracking each time an agent engages in a transaction, and retained centrally. But in addition, we would like to assess the honesty of the agent. Towards this aim, we ask that each agent declares its anticipated level of participation in the community before it joins and we determine whether the agent's reporting has been honest or not, based on what transpires once the agent is within the community. We refer to this as External Trustworthiness.

The aim of this approach is twofold: i) we would like to reward honest agents, even if their level of contribution is relatively light, and ii) we would like to identify

those agents that deserve more careful monitoring (those with suspect honesty), even if the quality of their contribution appears to be quite high.

2.1.2 Incentives for communities to share reputation ratings of agents

Consider now the context where there are multiple communities within a multiagent system, each with their own agents, some of which are well respected contributors and others which are merely free-riders, whitewashers, malicious agents or agents for which very little is yet known. Consider as well that agents have the ability to not only join new communities, but to leave from their current communities. It would appear that communities would try their best to retain their most valued agents and might even resort to untruths, in order to facilitate the exit of their less desirable agents.

In our framework, what we in fact need is almost the opposite to this situation. We would like communities to truthfully report the reputation rating of their agents, when asked. Because of this, we require some kind of incentive for the community to not only report fairly about their most valued agents but also to report fairly about their less desirable agents. The topics of mechanism design and game theory have been examined extensively by multiagent system researchers exploring the design of electronic marketplaces. An incentive mechanism is a method that brings rewards for honesty or penalties for dishonesty, in such a way that rational parties are inclined to be truthful (e.g. [7]).

Various researchers have proposed incentive mechanisms for P2P environments. A common approach is to design a utility function that charges agents for downloads and rewards them for uploads, including an amount of money to be paid for using the network or to receive as payment for contributing to the network (as in [8]). This is to motivate agents to be good contributors. For our research, we need an incentive mechanism to motivate communities to share information about their agents freely.

An incentive based approach to community sharing of reputation ratings

Example 1. Consider the agent a_0 which participates in communities c_A, c_B, c_C, and c_D with reputation 0.4, 0.6, 0.55, and 0.9, respectively. Consider now that a_0 decides to join other communities and it has made a list of the following candidate communities: c_F, c_G, and c_H. The communities c_F, c_G, and c_H have to decide if they will allow the agent a_0 to join them. In order to make this decision, they are interested in acquiring information regarding the agent's reputation, truthfulness and contribution in the communities the agent participated in since time t_0. Without loss of generality we assume that the agent did not participate in any other community than the ones in which it currently resides. Consider now that the community c_F wants to query the communities c_A, c_B, c_C and c_D regarding agent a_0's behavior.

In order to motivate the latter communities to exchange this information we consider a payment based system. Very briefly, the general idea is the following: when a community c_i wants to acquire information regarding an agent from another community c_j it has to pay some 'money' to the community c_j. The amount of money should depend on the degree that the agent is valuable as well as to the type of the community c_F. For instance, if the communities c_j and c_i are of similar type then the exchange of information between them might be more valuable than if they were of different types.

In this subsection, we sketch an incentive mechanism that could be used to motivate communities to provide truthful ratings about their most reputable agents. Our utility function is presented below:

$$U_{c_j}^{ExRp} = U_{c_j}^{Gain} + U_{c_j}^{Comp} - U_{c_j}^{Cost} - U_{c_j}^{Pnlt}. \tag{2.1}$$

The utility function $U_{c_j}^{ExRp}$ should increase monotonically with respect to the reputation of the agents for which the community c_j provides information. Also, it should monotonically decrease with respect to the agents' reputation each time the community c_j either acquires information by another community regarding the latter agents or each time it proved that it provided incorrect information regarding these agents. More specifically:

$U_{c_j}^{Gain}$: Represents the rewards the community c_j gained from providing advice about its agents. Currently we define:

$$U_{c_j}^{Gain} = \sum_{(c_k,a_i)\in PAd_j} rel(c_j,c_k) * rep_{c_j}^{a_i} \tag{2.2}$$

where $rep_{c_j}^{a_i}$ depicts the reputation of agent a_i inside the community c_j, $rel(c_j,c_k)$ represents the relevance between the communities c_j and c_k, where $1 \leq rel(c_j,c_k) \leq 10$, and PAd_j is a set that represents for which agents and to which communities the community c_j provided information. In particular, it consists of vectors (c_k,a_i) which represent the information that community c_j provided to community c_k regarding agent a_i. The reason why we also consider the reputation of the agent inside the community c_k is twofold: first it provides an incentive to communities to assist the agents to join communities that can better match their interests, and secondly it balances the drawback of providing information regarding reputable agents.

The relevance factor $rel(c_j,c_k)$ depicts the importance of the services that community c_j offers with respect to the community c_k. For example, consider two communities c_0 and c_1 which represent an e-marketplace and a file sharing community, respectively. If, for instance, c_0 provides information regarding the reputation of its agent a_0 to the community c_1 then this information may be more valuable information than if c_1 provided information regarding the agent a_0 (e.g. if the reputation of the agent in c_0 was 0.6 and in another peer to peer community

c_2 was 0.7, then for the community c_k the reputation in c_0 might encapsulate more valuable information than the reputation of the agent in the community c_1 despite the fact that it is higher). This may be true, for instance, because it may be harder to build up a high reputation in a marketplace setting, where there are typically fewer transactions.

$U_{c_j}^{Cost}$: Represents the cost the community c_j has to pay for requesting information regarding agents. Currently:

$$U_{c_j}^{Cost} = \sum_{(c_k,a_i)\in PAd_j} rel(c_j,c_k) * rep_{c_j}^{a_i} \tag{2.3}$$

where PAd_j is a set that represents for which agents and from which communities the community c_j requested information. In particular, it consists of vectors (c_k, a_i) which mean that community c_j requested information from community c_k regarding agent a_i.

The basic idea that underlines our utility function is the following: each time a community c_j provides information about an agent to a community c_k its utility function is increased by the reputation of the agent in community c_j. This provides incentive for communities to share information for highly reputable agents and of course regarding agents with low reputation there is always the incentive of getting rid of them. Each time a community c_j requests information about an agent from a community c_k it deducts from its utility function an amount equal to the reputation of the agent in the community c_k. And finally, each time a community provides incorrect information regarding an agent, its utiflity function is decreased by the same amount it gained for providing this information multiplied by a factor ω greater than one. The factor ω is increased with respect to the number of times the community provided incorrect information.

$U_{c_j}^{Pnlt}$: Represents the penalty the community c_j has to pay for providing false information about the reputation of its agents. Currently:

$$U_{c_j}^{Pnlt} = \omega_{c_j} * \sum_{(c_k,a_i)\in PrNAd_j} rel(c_j,c_k) * rep_{c_j}^{a_i} \tag{2.4}$$

where $rep_{c_j}^{a_i}$ depicts the reputation of a_i in c_j as it was calculated by the community c_j, and $PrNAd_j$ is set that represents for which agents and to which communities the community c_j provided incorrect information. Similar to PAd_j, $PrNAd_j$ consists of vectors (c_k, a_i) which means that community c_j provided information to community c_k regarding agent a_i. ω_{c_j}, $\omega_{c_j} : [0,1] \mapsto [1, e\theta]$, plays the role of a penalty factor that also has a fading property [1,2] and is calculated based on the following formula:

$$\omega_{c_j} = \begin{cases} 0 & n_f(c_j) = 0 \\ \theta * e^{\frac{n_f(c_j)}{n_{total}(c_j)}} & n_f(c_j) \geq 1 \end{cases} \tag{2.5}$$

where $n_f(c_j)$ is a parameter representing the number of times the community c_j provided incorrect information to other communities since time t_0, $n_{total}(c_j)$

represents the total number of times that community provided information to other communities since time t_0, and θ, is a parameter which depicts the severeness of the penalty for providing wrong information (currently we consider $\theta = 1$). The proposed formula exponentially increases the penalty that each community has to pay with respect to the number of times it provided inaccurate information but also gives a chance to communities with previous bad behaviour.

$U_{c_j}^{Comp}$: Represents the compensation the community c_j gains when it proved that it received wrong information from another community. This compensation is equal to the penalty that the community will have to pay for providing inaccurate information.

Example 2. Consider the community c_F in example 1 which is starting the procedure of reasoning whether to accept the agent a_0. In order to initiate the reasoning it would like to acquire information from the communities the agent participated in since time t_0. By using the approach we just presented, the utility function of the community c_F will decrease by 2.55 (=0.9+0.6+0.55+0.4) while the utility functions $U_{c_A}^{ExRp}$, $U_{c_B}^{ExRp}$, $U_{c_C}^{ExRp}$, and $U_{c_D}^{ExRp}$ of c_A, c_B, c_C, and c_D will increase by 0.4, 0.6, 0.55, and 0.9, respectively. Assume now that somehow it is shown that the information the community c_A provided was inaccurate and it is the first time that provided inaccurate information, so $n_f(c_A) = 0$, and that the total number of times that it has provided information up to this point is 49, so $n_{total}(c_A) = 49$. In addition we set $\omega = 1$. The values of $n_f(c_A)$ and $n_{total}(c_A)$ will be increased by 1, and thus $\omega = e^{\frac{n_f(c_A)}{n_{total}(c_A)}} = e^{0.02}$. The utility function of the community c_A will be decreased by $U_{c_A}^{Pnlt} = 0.40808 (= 0.4 * e^{0.02})$, which equals the amount it acquired, $U_{c_A}^{Pnlt} = 0.4$, plus a penalty of 0.00808, while the utility function of c_F will increase by $U_{c_F}^{Pnlt} = 0.40808$.

Consider now that the community c_C also proved to provide inaccurate information, but this community has a history of providing inaccurate information with $n_f(c_C) = 14$ and $n_{total}(c_A) = 44$ since time t_0. This will result in $n_f(c_C) = 15$ and $n_{total}(c_C) = 45$, which means that c_C's utility function will be decreased by $U_{c_C}^{Pnlt} = 0.74242 (= 0.55 * e^{0.3})$ and in this case the penalty for c_C is 0.192422. As in the previous case, the utility function of c_F will be increased by $U_{c_F}^{Pnlt} = 0.74242 (= 0.55 * e^{0.3})$.

The above incentive mechanism however requires a method for detecting whether a community provided inaccurate information regarding an agent. Since the problem of accurately detecting if a community lied is very difficult, we are looking into ways for solving it indirectly, in other words, by finding a way to provide strong incentives for the communities not to lie in the first place.

An auction-based approach for community sharing of reputation ratings

In this subsection, we outline an alternative to the incentive mechanism presented in Section 2.2.1 that still incorporates some of its elements. More specifically, we consider the use of a Vickrey auction [7], also known as second price sealed bid

auction. The way a Vickrey auction works is the following: the winning bidder is the one with the highest bid but the price it has to pay is equal to the second highest bid. The main reason we have chosen in our first steps the use of a Vickrey auction is because the Vickrey auction is incentive-compatible (i.e. no agent has the incentive to not submit a bid that reflects the true value of the good), thus it promotes honesty between both the auctioneer and the bidder. Despite the traditional problems that this type of auction has in regards to the existence of malicious auctioneers that try to deceive the bidders by placing fake bids in order to increase the second highest bid and thus the paid price, in our research we need to deal with cases where the auctioneer does not really want to sell an item (e.g. information regarding the reputation of a very reputable agent) and consequently tries to overprice it with the hope that no agent will submit. Similar to [11] we will consider a Vickrey auction with reserve prices in order to prevent coalitions of bidders from manipulating the auctions. Each community c_j has a utility function $U_{c_j}^{ExRp}$ and can act as both auctioneer and bidder. It acts as auctioneer when it is interested in increasing its utility $U_{c_j}^{ExRp}$ by providing information regarding its agents, and as a bidder when it is interested in utilizing its $U_{c_j}^{ExRp}$ for acquiring information regarding candidate agents. The goods that each auctioneer offers is the information it has regarding the reputation of an agent. At the moment we consider that the price of the good is equal to the reputation of the agent. A community might decide to offer multiple copies of a good. By giving this flexibility to a community, we allow it to protect itself from losing reputable agents.

In particular, we consider the following rules that each community has to follow when it acts as an auctioneer:

- Rule No 1: Each community should provide at least one copy of information for each of the agents it hosts, while the total number of copies for non-reputable agents has to be equal to the total number of copies for reputable agents.
- Rule No 2: The local reputation of an agent a_i inside the auctioneer's community defines the lowest price of the second highest bid that the auctioneer allows in order for an auction not to be canceled. This price is known as the 'reserve price'.

The first rule aims to motivate the communities not to hide their highly reputable agents, while the second rule prevents bidders from forming coalitions to acquire the goods at very low prices. Please note that in a Vickrey auction (which does not use reserve prices) the bidder who offers the highest bid will acquire the good. This means that even if the actual price of the good was 1000, and the two highest bids where 1 and 0.5 then the auctioneer will have to give out the good and it will only get paid 0.5. The way our game works is as follows: Each community c_j for each of its agents a_i provides a trusted entity E with tuple $(a_i, rep_{a_i}, cop_{a_i})$ which represents the following information: the community c_j can provide information about the agent a_i to cop_{a_i} number other communities. The reserve price it considers is equal to rep_{a_i}.

The entity E initiates a Vickrey auction. Please note that for simplicity reasons initially we consider that cop_{a_i} is equal to 1. Each community c_k which is interested in acquiring information from the community c_j for an agent a_i will have to submit a bid b_{c_k}. The community c_l which submitted the highest bid will win the auction only and only if $b_{c_l} > rep_{a_i}$. In case where the second highest bid was less than the reserve price then the winning community will pay an amount equal to the highest bid. If both the highest bid and the second highest bid were below the reserve price then the auction will be canceled and re-initiated.

Example 3. Consider the community c_D in example 1. It will run a Vickrey auction in which the communities c_F, c_G and c_H will participate as bidders. Assume now that the community c_D is very concerned regarding releasing the information of the agent a_0 since it is a very reputable agent. For this purpose, it is interested in only providing one 'copy' of information. Consider now that the communities c_F, c_G and c_H have put bids 0.91, 0.95, and 0.5, respectively. The community, which wins the auction and thus will get information regarding agent a_0, is the community c_G. The price it will have to pay is 0.91, and thus its utility function should be decreased by 0.91 while the utility function of the community c_D will be increased by an equal amount.

2.1.3 Interpreting ratings provided by communities

Once communities have an incentive to share the reputation ratings of their agents, there is a need to interpret the ratings that are provided, in order to make use of the information to decide which new agents to accept. We first of all envisage a procedure hereby subjective differences of other communities is considered. For example, the approach of Regan et al. [6] allows the evaluation function of an entity to be learned, over time, using Bayesian reasoning, in order to make use of the information that is provided.

Another consideration is the trustworthiness of the community and this can also be modeled, over time, using methods motivated by various trust and reputation models (e.g. [1–3, 6]).

We in fact propose a procedure whereby the agent desiring to join the community will self-report its reputation in previous communities; this can then be compared against the report provided by these communities in order to judge the trustworthiness of the agent and of the community. In particular:

1. The community c_{cand} asks the agent a_i for the list of the communities C_{a_i} it participated in since time t and its contribution in each of these communities. (Restricting to time t is to keep more relevant information).
2. The community c_{cand} asks the communities in C_{a_i} regarding the reputation and the contribution of the agent a_i, their population and the type of services they offer to the agents.
3. For each community $c_p \in C_{a_i}$ the community c_{cand} evaluates the information it received from both the agent and the community c_p (cross validation) by

checking if the contribution the agent reported is the same as the one that was reported by the community c_p.

a) If it is, then the community c_{cand} updates the importance of the agent a_i based on the information it received from the communities in C_{a_i}.

b) If it is not the same, then the community c_{cand} has to decide whether the agent is malicious or the communities lied and update accordingly the C_{a_i}.

i) If a community proved to report incorrect information regarding the agent's contribution, then it should be penalized.

ii) If the agent proved to report incorrect information regarding its contribution in the communities it participated in, c_{cand} and will deny access to the agent.

4. The community c_{cand} decides whether to accept an agent or not based on C_{a_i}.

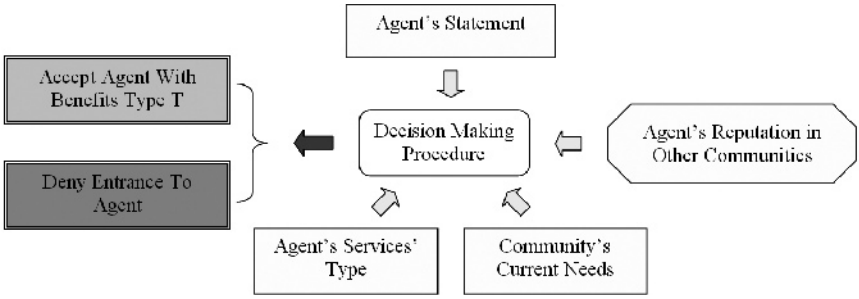

FIGURE 1. Community reasoning.

2.1.4 Overview of community reasoning procedure

Figure 1 presents an overview of the reasoning a community undertakes in order to decide whether to accept an agent:

1. Gets the agent's history h_{a_i}- a record of the communities in which the agent has participated.
2. Decides which communities reported in h_{a_i} to consult.
   a) Evaluates what it can afford to pay other communities
   b) Considers the degree of trustworthiness of each of the above communities.
3. Contacts the communities to acquire information about the agent.

4. Initiates the decision making procedure (Figure 1). The decision procedure takes as input:
   a) The type of the services the agent can contribute to the community.
   b) The current needs of the community (e.g. resources that the community needs etc)
   c) The statement of the agent regarding its anticipated contribution to the community.
   d) The reputation of the agent in the communities it participated.
5. Decides whether to accept the agent or not.

2.2 Agents reasoning about communities

In order for an agent to determine the best communities to join, it is important to evaluate: i) the trustworthiness of the community, and ii) the utility that can be gained by participating in the community.

Whether it is possible to track the trustworthiness of a community, when reporting about its agents, presents some challenges. As discussed, if it is possible to detect and record each time a community is untruthful about the reputation of one of its agents and this information becomes shared knowledge within the community, then agents will have an avenue of learning about the most reliable communities to join (in order to enjoy a good reputation, with them). But one issue to address is the fact that inaccurate trustworthiness ratings from malicious agents within the community can cause the inaccurate reporting of the reputation of the agent.

An agent can however reason with the advertised utility function of the community, to determine the potential benefit of joining this community. This assumes that each community would advertise a type of utility function for its agents. For example, a P2P system might propose a utility function such as [8]:

$$U_i = \left[f_i^{AD}(AD) + f_i^{NV}(NV) + f_i^{AL}(AL)\right] - \left[f_i^{DS}(DS) + f_i^{BW}(BW)\right] - FT \quad (2.6)$$

where AD represents the amount downloaded, NV the network variety, DS the disk space used, BW the bandwidth used, AL the altruism (in case the agent derives utility from the satisfaction of contributing to the network), and FT represents the financial transfer (i.e. the amount of money the agent has to pay for using the network or the amount of money it might get paid for contributing to the network).

In addition, we propose to model and share the average utility value of the agents currently residing within a community. More specifically, we consider that the agent a_i ranks the communities based on a parameter. This parameter measures the expected utility of the agent with respect to the average utility that the other agents in c_p have. The formula for calculating this value is:

$$gcop_{c_p}^{a_i} = \frac{E[U_{c_p}^{a_i}]}{AVG(U_{c_p})} \quad (2.7)$$

where $E[U_{c_p}^{a_i}]$ is the expected utility in c_p that the agent a_i will gain if it joins the community c_p, and is the average utility of the agents in c_p. In our formula we consider the average utility of the agents inside the community mainly for one reason: it can give us a better idea of the reputation as well as the benefits the agent can get by entering the community. For example, knowing that the expected utility an agent gains by joining a community is 5 does not provide any kind of information but we can somehow know that the average utility in that community is 2, then this means that the agent can gain utility which is 2.5 times the average utility the other agents have. Obviously this encapsulates more valuable information than simply calculating the expected utility value.

2.3 Privacy considerations

In order for the framework presented in Sections 2.1 and 2.2 to be effective, we require a method for addressing concerns for privacy that agents may have. In particular, we have two proposals.

The first is to allow agents to ascribe to one of three main categories: 'privacy sensitive' or 'privacy concerned' or 'privacy disinterested'. Agents that are 'privacy sensitive' do not allow any community to share any kind of information about them. In contrast, agents that are 'privacy disinterested' allow the communities to exchange freely information regarding them (e.g. reputation, participation), while agents that are 'privacy concerned' are willing to compromise their privacy if they can gain significantly higher profit. Similarly, we consider that a community can be either 'privacy interested' or 'privacy disinterested'. In the first case, the community does not share the information of any of its agents, even if the agent is not privacy sensitive, while in the second case the community can share the information of any privacy disinterested agent with any other community.

Once there is the above categorization for agents an agent can increase its utility by forgoing its privacy. We are also aware that when information is shared between communities, the privacy of an agent may be compromised even further than is immediately suggested. For example, a major challenge in privacy preservation is to ensure that there is no unintentional leak of private information. Consider an agent that belongs to several communities and it is going to reveal this information to a new community. Assume now that one of these communities is related to patients with chronic diseases, and the other is specialized in low sugar products. The knowledge of the participation of the agent in both of the communities can lead to the inference that the agent represents a diabetic user, an information that the user might not wish to freely share with others.

In ubiquitous computing environments research has been done [4,5] to develop a theoretical approach that makes use of policies to determine who is allowed to access to information and at what level of granularity. This is one possible method for addressing concerns about sharing private information inadvertently.

For our framework, we are interested in exploring how privacy can be protected as part of the incentive mechanism for sharing information between the

communities. Communities need to recognize that agents may want to keep part of their information private.

3. Discussion

In this paper, we presented a framework for communities of intelligent agents to share reputation ratings of their agents. The aim was to promote effective community structure within multiagent systems, allowing agents to derive benefits from participating in communities that consist of trustworthy, good contributors and allowing communities to be more effective in providing a beneficial environment that will be attractive to newcomers.

Although the framework is generally applicable to any multiagent system that can model the trustworthiness of its agents (in terms of both quality of contribution and degree of participation), we have clarified its usefulness in the context of P2P file sharing systems, where agents typically both upload and download content, within their communities. Several other researchers have been examining incentive mechanisms for P2P networks, in order to encourage honest and productive behaviour from their agents [8,12–16]. Many of these approaches are utility-based and within a monetary context, where agents gain or lose units of value, based on their participation and contribution. In contrast, our work aims to provide incentives for communities to share information about their agents. It is expected that these communities would still be promoting honest behaviour from their agents, with other incentive mechanisms. But with the reputation of agents being shared between communities, there is now an additional incentive for an agent to be a good contributor within its current communities – in order to derive benefits when interested in joining a new community.

We have also outlined some approaches that can be used both by communities in their selection of agents and by agents in their selection of communities. This is towards an effective overall community structure within the multiagent system, where the utility of agents and of communities is maintained at a desirable level.

We have two key aims: i) to exploit the information that each community accumulates by exchanging it with other communities not only directly but also indirectly, putting pressure on the agents by informing them that any misbehavior will be spread to the communities they might wish to join in the future, and ii) to prevent agents from overstretching themselves among many communities, since this may lead them to provide very limited services to each of these communities due to their limited resources.

Our aim is to promote this idea by presenting an incentive based approach that will motivate the communities to share the knowledge they have built to the extent where their personal interests are compromised the least, while the agents, which are aware of this procedure, will consider more carefully their decisions regarding joining new communities. To the best of our knowledge the problem of designing a mechanism that allows communities to cooperate for exchanging

information regarding the reputation of the participant agents with respect to privacy constraints is a novel one.

4. Future work

Although various elements of our proposed framework are currently in a preliminary stage, we feel there is merit in presenting the overall framework as a novel approach to promote effective community structure in multiagent systems. The ultimate aim is to have communities with the best agents and agents residing in the best communities, so that the overall structure of the multiagent system is optimized. Trying to enable both agents and communities to achieve optimal utility at the same time is a major challenge. Considering that the related problem of determining the optimal coalition structure in a multiagent system is NP-hard [10], we will likely have to design algorithms that provide high utility to both agents and communities, without necessarily proving optimality. Addressing the balance between the needs for agents and communities is a topic for future research.

There are several other directions for future research. In particular, we would begin by examining in more detail some of the stronger assumptions within our proposed solution: i) that each agent is willing to report its expected contribution to a community, in order to calculate external trustworthiness (per Section 2.1.1) ii) that information regarding the average utility that agents have inside a community can be effectively modeled and shared (per Section 2.2). We also need to address within our proposed auction-based mechanism for sharing reputation ratings between communities the concern that an auctioneer might overprice an item because it does not want to sell it.

To extend our proposal for agents to reason about communities, agents might be interested in modeling the way different types of communities model their utility function or the reputation they assign to their participants. Since this is a problem with high uncertainty due to the information agents may have, a probabilistic approach (such as in [2,6]) appears to be a promising starting point.

In addition, agents need to reason about which communities they will attempt to join, concurrently. This suggests introducing techniques such as optimization, in order to model this kind of reasoning.

As for extending the proposal for communities, even if we solve the problems that emerge when exchanging information, the next crucial step is the exploitation of this information for evaluating the candidate agents. An open question is how best to implement the procedure for comparing the agent's self-reported rating with that provided by its community. Since different communities may use different ways of evaluating the reputation of their agents (e.g. [3,9]), it would be interesting to study whether an approach similar to [6] which learns the evaluation function of an agent can be used as a starting point to provide accurate and efficient results.

References

[1] T. Tran and R. Cohen, *Improving user satisfaction in agent-based electronic marketplaces by reputation modelling and adjustable product quality*, AAMAS'04: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems (Washington, DC, USA), IEEE Computer Society, 2004, pp. 828–835.

[2] A. Josang and R. Ismail, *The beta reputation system*, In Proceedings of the 15th Bled Electronic Commerce Conference, Bled, Slovenia, 2002.

[3] J. Zhang and R. Cohen. *Towards more effective emarketplaces: A novel incentive mechanism* In Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'07) Workshop on Trust in Agent Societies, 2007.

[4] U. Hengartner and P. Steenkiste, *Access control to information in pervasive computing environments*, HOTOS'03: Proceedings of the 9th conference on Hot Topics in Operating Systems (Berkeley, CA, USA), USENIX Association, 2003, pp. 27–27.

[5] U. Hengartner and G. Zhong, *Distributed, uncertainty-aware access control for pervasive computing*, PERCOMW '07: Proceedings of the Fifth IEEE International Conference on Pervasive Computing and Communications Workshops (Washington, DC, USA), IEEE Computer Society, 2007, pp. 241–246.

[6] K. Regan, P. Poupart, and R. Cohen, *Bayesian Reputation Modeling in E-Marketplaces Sensitive to Subjectivity, Deception and Change*, International Conference on Machine Learning , 2006.

[7] S. Russell and P. Norvig, *Artificial intelligence: A modern approach*, 2nd ed., Prentice-Hall, 2003.

[8] P. Golle, K. Leyton-Brown, and I. Mironov, *Incentives for sharing in peer-to-peer networks*, EC '01: Proceedings of the 3rd ACM conference on Electronic Commerce (New York, NY, USA), ACM Press, 2001, pp. 264–267.

[9] A. Josang, R. Ismail, and C. Boyd, *A survey of trust and reputation systems for online service provision*, Decis. Support Syst. **43** (2007), no. 2, 618–644.

[10] T. Sandholm, K. Larson, M. Andersson, O. Shehory, and F. Tohme, *Coalition structure generation with worst case guarantees* Artificial Intelligence, 1999, 111(1–2), pp. 209–238.

[11] L. M. Ausubel, and P. Cramton, *Vickrey auctions with reserve pricing*, Tech. report, Economic Theory, 1999.

[12] K. Lai, M. Feldman, I. Stoica, and J. Chuang, *Incentives for cooperation in peer-to-peer networks*, 2003.

[13] R. T. B. Ma, S. C. M. Lee, J. C. S. Lui, and D. K. Y. Yau, *Incentive p2p networks: A protocol to encourage information sharing and contribution*, SIGMETRICS Perform. Eval. Rev. **31** (2003), no. 2, 23–25.

[14] R. T. B. Ma, S. C. M. Lee, J. C. S. Lui, and D. K. Y. Yau, *An incentive mechanism for p2p networks*, ICDCS '04: Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS'04) (Washington, DC, USA), IEEE Computer Society, 2004, pp. 516–523.

[15] J. Vassileva, and R. Cheng, *Adaptive reward mechanism for sustainable online learning community*, In Proc. in Education, AIED 2005, pp. 152–159.

[16] B. Yu, and M. P. Singh, *Incentive mechanisms for peer-to-peer systems*, In In Proceedings of the Second International Workshop on Agents and Peer-to-Peer Computing, 2003, pp. 77–88.

[17] G. Weiss, *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. MIT Press, Cambridge, MA, USA, 1999.

Georgia Kastidou
David R. Cheriton School of Computer Science
University of Waterloo
Waterloo, ON, N2L3G1
Canada
e-mail: gkastidou@cs.uwaterloo.ca

Robin Cohen
David R. Cheriton School of Computer Science
University of Waterloo
Waterloo, ON, N2L3G1
Canada
e-mail: rcohen@uwaterloo.ca

Economic Models and Algorithms for Distributed Systems, 63–81
Book Series: Autonomic Systems

Formation of Virtual Organizations in Grids: A Game-Theoretic Approach

Thomas E. Carroll and Daniel Grosu

Abstract. The execution of large scale grid applications requires the use of several computational resources owned by various Grid Service Providers (GSPs). GSPs must form Virtual Organizations (VOs) to be able to provide the composite resource to these applications. We consider grids as self-organizing systems composed of autonomous, self-interested GSPs that will organize themselves into VOs with every GSP having the objective of maximizing its profit. We formulate the resource composition among GSPs as a coalition formation problem and propose a game-theoretic framework based on cooperation structures to model it. Using this framework, we design a resource management system that supports the VO formation among GSPs in a grid computing system.

Mathematics Subject Classification (2000). Primary 91A40, 68M14, 68T99; Secondary 91A10.

Keywords. Grid computing, virtual organization, self organization, cooperative game theory

1. Introduction

Grid computing is *the* preferred computational platform of choice for collaborative, resource intensive applications which are typical in the domains of science and engineering [1]. There are two classes of participants in the grid, the users and the service providers. The users submit applications to be executed. The service providers, who are geographically distributed over the world, provide and operate resources to execute programs. Each provider has its own administrative domain and thus it is largely autonomous.

This research was supported in part by NSF grant DGE-0654014.

Applications require the composition of resources owned by several Grid Service Providers (GSPs). Service providers will form Virtual Organizations (VOs) [1] to provide the necessary resources on a *dynamic, per application basis*. The VO may dissolve as soon as the application has completed execution. Execution results in the service providers incurring costs (e.g., power, administrators wages and time). If we assume that the service providers are *rational* (self-interested and welfare-maximizing), they will refuse to offer their resources unless they can recover their costs.

The rationality assumption permits analysis by economic models. Of particular interest are *coalitional games*, which are studied in game theory. Coalitional games model the interactions between groups of decision-makers. In this case the decision-makers are the service providers. The service providers form VOs in such a way that each provider maximizes its own profit. The VOs provide the composite resource needed to execute applications.

A VO is traditionally conceived for the sharing of resources, but it can also represent a business model [1]. In this work, a VO is a coalition of GSPs who desire to maximize their individual profits and are largely indifferent about the global welfare.

In this paper we examine VO formation using models from coalitional game theory. We assume that a grid user submits a program and a specification consisting of a deadline and payment. A VO will form and execute the program. If the VO completes the program before the deadline, the VO will be paid; otherwise, the VO incurs the cost of execution. A GSP attempts to form a VO with other GSPs that maximizes its profit. But this is not simple as for n service providers, there are 2^n potential VOs in which a GSP can be part of. Further complicating the matter is that computing the worth of a coalition of GSPs requires determining a mapping that assigns application tasks to providers. This problem is known to be NP-hard. This process clearly takes an exponential amount of time and it makes completing the application before its deadline difficult. Consequently, a service provider cannot possibly compute the worth of all coalitions and, thus, has limited information to base its decision. It is likely that service providers will employ heuristics to expedite the process. These heuristics would be held private as they give providers significant economic advantages over the others. The advantage is only gained though, if others can be convinced to form the preferred coalition. This can be readily done by disclosing the coalition and its mapping.

Using the model that we developed, we propose a *resource management system*, the Virtual Organization Formation Manager (VOFM), to support VO formation among GSPs. The system is a middleware component which is supported by the grid implementation. The VOFM would be replicated across the grid, allowing users and GSPs to interact with local copies. The VOFM provides support for exchanging mappings and other information between GSPs to facilitate the

VO formation process. Out of the set of resulting VOs, it selects the one that will execute the program.

1.1 Our contributions

We model the Virtual Organization (VO) formation as a coalition formation problem. We use Myerson's cooperation structures [2] as the underlying coalition formation model. Coalition formation requires determining the value of various coalitions. GSPs will employ heuristics in determining these values. We design one such heuristic that can be used by the GSPs. It is likely that GSPs will derive different heuristics that benefit themselves and consequently, they would be unwilling to share them. We examine this situation and show that GSPs can benefit from different heuristics if they share the results. Finally, we propose a resource management system that facilitates VO formation by providing the necessary structures for GSP information control and exchange.

1.2 Related work

An important research topic in grid computing is the formation of virtual organizations. A *Virtual Organization* (VO) is an alliance among GSPs to collaborate and to pool their resources to compute large scale applications [1]. VO requirements for a grid architecture are discussed in [3]. Dynamic VO formation among autonomous agents and the management of such VOs are examined in [4]. VO formation in the CONOISE project is described in [5]. Furthermore, [6] describes the mechanisms for supporting dynamic VO formation and operation in CONOISE-G. VO formation requires resource discovery, which is a time consuming process. In [7], agents offering similar resources or composite resources form *communities* that are registered with the resource discovery service, thus, minimizing discovery costs. Coalitional game theory can be used to model VO formation among GSPs. *Coalitional game theory* examines the interactions between groups of decision makers. A good reference for coalitional game theory is [8]. One topic of interest is coalition formation, the partitioning of players into disjoint sets. There are many models of coalition formation (e.g., [9, 10]), but we are particularly interested in Myerson's cooperation structures [2]. Much research on coalition formation has been conducted in the multi-agent systems area for wide assortment of problems including distributed artificial intelligence [11] and service composition [12]. Further research has been conducted on task allocation and resource composition. Shehory and Kraus [13] proposed a distributed decision processing system in which the processing agents partition themselves into subsets with the goal of minimizing the ratio between coalition cost and coalition size. A taxonomy and classification of task allocation problems is given in [14]. For each class, a welfare-maximizing algorithm is proposed and its complexity analyzed. In [15], agents exchange idle time for the purpose of compositing resources to execute tasks. A simple non-game theoretic approach to coalitional formation in computational grids is proposed in [16].

Finally, machine learning techniques are employed to optimize coalition efficiency in [17]. Non-cooperative game theory was used to study the problem of scheduling tasks on parallel machines in [18, 19]. Several researchers have applied market-based approaches to resource allocation in grid computing [20–23]. Grosu and Das [24] investigated the benefits of various types of auction-based allocation protocols. Grosu [25] proposed an architecture for strategyproof resource allocation in grids.

1.3 Organization

This paper is structured as follows. In Section 2., we introduce the main concepts of coalitional game theory. In Section 3., we define the model that is employed throughout the paper. In Section 4., we propose a framework and a model of coalition formation based on cooperation structures. In Section 5., we propose a resource management system to support VO formation in grid computing based on the proposed coalition formation framework. Finally, we draw conclusions and present future directions in Section 6..

2. Coalitional game theory

In this section we introduce the main concepts of coalitional game theory that are employed in this paper. *Coalitional game theory* studies the interactions between groups of *decision-makers* (the *players*).

A coalitional game comprises a set of players, N of cardinality n. Every subset S of N, where S is a coalition, has a *value* or *worth* given by the *characteristic function* $v(S)$. The value can be thought of as the profit obtained when the members of a coalition work as a group. We formally define a coalitional game as follows.

Definition 2.1 (Coalitional game[1])**.** A *coalitional game* (v, N) is characterized by the *player set* N and the *characteristic function* $v : S \subseteq N \rightarrow \mathbb{R}^+$ such that $v(\emptyset) = 0$.

An important problem in coalitional game theory is coalition formation. *Coalition formation* is the partitioning of the players into disjoint sets. The set of coalitions, $\mathcal{S} = \{S_1, S_2, \ldots, S_k\}$, is such that each player is a member of exactly one coalition, i.e., $S_i \cap S_j = \emptyset$ for all i and j where $i \neq j$ and $\bigcup_{S_i \in \mathcal{S}} S_i = N$.

There are several models of coalition formation. One model is *coalitional structures* [9]; another model is *cooperation structures* [2, 26], which we discuss in detail in Section 4.. The coalition formation under cooperation structures is represented as an extensive-form game. In an *extensive-form game* (or simply *extensive game*), the players take turns playing the game.

[1] Note that we use the relaxed definition of a coalitional game which does not require superadditivity. Superadditivity is the property that for two disjoint sets R and S of N, $v(R \cup S) \geq v(R) + v(S)$.

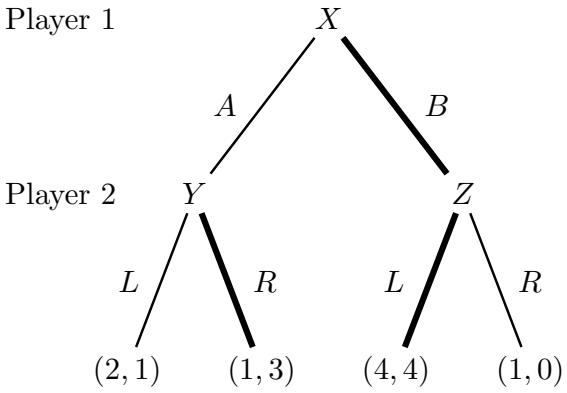

FIGURE 1. A 2-player, 2-turn game with the actions that are part of the subgame perfect equilibria represented by thick lines.

An extensive game can be modeled as a topological tree where the interior nodes are *turns*, the edges are *actions*, and the leafs are the *outcomes* of the game. The outcomes are represented as tuples; the i-th entry of an outcome is Player i's profit. The move of the first player is depicted at the root of the tree. As an example, Figure 1 depicts a 2-player, 2-turn game, where Player 1 moves first, followed by Player 2. Player 2 can observe the action of Player 1 (thus, this game is of *perfect information*). To solve such games, we use the concept of subgame perfect equilibria. Nodes X, Y, and Z are *subgames* of the complete game depicted in Figure 1. A solution is a *subgame perfect equilibrium* when no player can do better by changing her action for any subgame, when all the other players' actions are unchanged. Player 2 must choose either L or R for each of the subgames Y and Z. For subgame Y, she chooses R and for Z, she chooses L which are the actions that maximize her profit. Player 1 knows which actions Player 2 will choose due to the rationality assumption. Therefore, she chooses action B which leads to a profit of 4 units instead of A which only achieves 1 unit of profit. We have identified the actions that are subgame perfect by thick, solid lines in Figure 1. The procedure we used to determine the subgame perfect equilibria is called *backward induction*. Let the *length* of the subgame be the largest number of actions needed to reach a leaf. By convention, the length of terminal subgames (nodes Y and Z in Figure 1) is one. Backward induction computes the equilibria for subgames of length ℓ to compute the equilibria of games of length $\ell + 1$. Backward induction gives a set of strategy profiles. A *strategy profile* gives the action taken for each and every subgame in an extensive game. Each strategy profile determines an outcome, but different strategy profiles need not determine unique outcomes.

In the next sections we present a model for grid computing comprising autonomous service providers and show how the service providers will form VOs to execute programs.

3. Model

A user has an application program $\mathcal{T} = \{T_1, T_2, \ldots, T_n\}$ comprising n independent tasks that she desires to have executed to completion before deadline d. A set of grid service providers, $\mathcal{G} = \{G_1, G_2, \ldots, G_m\}$, provides resources for executing programs. We assume that each service provider is autonomous and it behaves rationally (self-interested and welfare-maximizing). Furthermore, we assume that each service provider G_j, for $j = 1, 2, \ldots, m$, owns and operates a single machine. The machines owned by GSPs form a network of heterogeneous, unrelated machines. The cost function, $c : \mathcal{T} \times \mathcal{G} \rightarrow \mathbb{R}^+$, gives the cost of executing each task $T \in \mathcal{T}$ when assigned to each service provider $G \in \mathcal{G}$. Additionally, the function $t : \mathcal{T} \times \mathcal{G} \rightarrow \mathbb{R}^+$ gives the execution time of each task when assigned to each GSP. It is assumed that once a task is assigned to a GSP it is executed to completion on the machine owned by that GSP, i.e., the task is neither preempted nor migrated. Techniques such as code profiling [27] and statistical prediction [28] can be used to estimate task execution times. The user submits the program, deadline d, cost function c, and execution time function t to the grid resource manager.

The user experiences an increase in welfare, ΔW, for having the program completed. She is willing to pay P, such that $P \leq \Delta W$, if the program is executed to completion by deadline d. If the program execution exceeds d, the user does not experience a welfare increase (i.e., $\Delta W = 0$) and thus, is not willing to pay any amount (i.e., $P = 0$) as she is rational.

As was stated above, GSPs incur cost for executing tasks. Service providers form VOs in order to have the necessary capacity to execute the program and more importantly, maximize their profits. The profit is simply defined as the difference between payment P and execution costs. In our model we represent VOs as coalitions of GSPs.[2] For each coalition S, where $S \subseteq \mathcal{G}$, there exists a mapping $\pi_S : \mathcal{T} \rightarrow S$, which assigns task $T \in \mathcal{T}$ to service provider $G \in S$. The costs incurred for executing the program $\mathcal{T}$ on S under mapping π_S is given by

$$C(\mathcal{T}, S) = \sum_{T \in \mathcal{T}} \sum_{G \in S} \sigma_S(T, G) c(T, G) \tag{3.1}$$

where $\sigma_S(T, G)$ is an indicator such that

$$\sigma_S(T, G) = \begin{cases} 1 & \text{if } \pi_S(T) = G \\ 0 & \text{if } \pi_S(T) \neq G\,. \end{cases} \tag{3.2}$$

The execution time of the program is given by its *makespan* (i.e., completion time) as induced by the mapping π_S. The execution time is given by

$$E(\mathcal{T}, S) = \max_{G \in S} \sum_{T \in \mathcal{T}} \sigma_S(T, G) t(T, G)\,. \tag{3.3}$$

[2]Since coalitions have a well defined meaning in game theoretical modeling we will use the term coalition instead of VO when we describe the theoretical model of VO formation.

Each GSP has the objective of determining the coalition in which it is a member and where it will receive the greatest profit. The money earned by the coalition is divided in some fashion among its members. Even if a coalition has low costs, it may be unattractive due to its low profit margins.

We formalize this scenario as a coalitional game (v, N), where the player set N is the set of GSPs $\mathcal{G}$. The characteristic function v is then defined as

$$v(S) = \begin{cases} 0 & \text{if } |S| = 0 \text{ or } E(\mathcal{T}, S) > d \\ P - C(\mathcal{T}, S) & \text{if } |S| > 0 \text{ and } E(\mathcal{T}, S) \leq d\,, \end{cases} \tag{3.4}$$

where $|S|$ is the cardinality of S. Note that $v(S)$ satisfies the constraint $v(\emptyset) = 0$.

Now the question is how the profit will be divided among the members of a coalition. Traditionally, the *Shapley value* [29, 30] would be employed, but computing the Shapley value requires iterating over every partition of a coalition, an exponential time endeavor. If we assume that all members remain in the coalition until the program is completed, the *equal sharing* of the profit among members seems justifiable. Equal sharing has been successfully used in other systems where tractability is critical (e.g., [13]).

As we have stated above the welfare-maximizing GSP will determine its coalition by considering the profit it earns and not the coalition value. Therefore, a service provider G determines its preferred coalition S by solving:

$$\max_{(S)} \frac{P - C(\mathcal{T}, S)}{|S|} \tag{3.5}$$

subject to:

$$E(\mathcal{T}, S) \leq d \tag{3.6}$$

and

$$G \in S\,. \tag{3.7}$$

In the above, (3.5) is G's profit maximizing objective subject to the constraints that the program is completed before the deadline (3.6), and that G is a member of the coalition (3.7).

In this section we discussed the motivations and presented the model for coalition formation. Next, we discuss the VO formation process.

4. Virtual organization formation

In this section we investigate the virtual organization formation process considering the coalitional game model proposed in Section 3.. A GSP must choose which VO to form (or it can decide to form a VO consisting just of itself). Since the GSP

is self-interested, it will choose to form a VO where it maximizes its profit. As in the previous section, we will use the term coalition when we refer to VO.

Computing the best coalition S requires determining the mapping π_S. Assume for the moment that determining the mapping is tractable. Service provider G computes its profit for each of the potential 2^{m-1} coalitions where it can be a member and then chooses the coalition S that maximizes its own profit, i.e., G's preferred coalition is $S = \arg\max_{S \subseteq \mathcal{G}} v(S)/|S|$. Note that there may exist coalitions such as S' with greater values than G's choice, but these coalitions do not maximize G's profit, i.e., $v(S) < v(S')$ but $v(S)/|S| > v(S')/|S'|$. Determining a preferred coalition is further complicated by the fact that coalition formation is a non-cooperative process. The following example illustrates why. Let G determine its preferred coalition as S and let S contain GSP G'. Similarly, G' determines its preferred coalition as S', but S' does *not* contain G as a member. The question is will G and G' be members of the same coalition or will they be members of different coalitions? These issues are resolved by the appropriate coalition formation model.

One model of coalition formation is *coalition structures* [9]. Another approach to coalition formation and the one we choose, is based on Myerson's *cooperation structure* (or *cooperation graph*) [2, 26]. In the cooperation structure model coalitions form by *building links* between pairs of GSPs. A *link* signifies that the service providers can carry on direct negotiations within a coalition. When instantiating a link, both GSPs must agree to the link or the link is not created. The link outcome is announced to all. Once a link is formed, it cannot be broken (the links are enforced by binding contracts). As the links are formed each GSP updates what it perceives to be its best coalition.

Before link building commences, an *order of rule* is announced that specifies the order in which links can be built. We are assuming a fully connected network, thus the order of rule contains the $n(n-1)/2$ possible links. Another consequence of the fully connected network is that each member of a coalition will have a link to every other member of that coalition [26].

We now provide a simple example of coalition formation. Let $\mathcal{G} = \{G_1, G_2, G_3\}$ and the resulting $v(S)$ be

$$v(S) = \begin{cases} 0 & \text{if } |S| \leq 1 \\ 20 & \text{if } |S| = 2 \\ 24 & \text{if } |S| = 3\,. \end{cases} \tag{4.1}$$

The order of rule is $\{G_1\,G_2, G_2\,G_3, G_1\,G_3\}$. The link $G_1\,G_2$ is decided first. We examine the case in which GSP G_1 and G_2 form a link, each obtaining 10 units of profit. If G_2 builds the link to G_3, its profit will correspondingly reduce from 10 to 8 units. Thus, G_2 will reject forming this link. Similarly, G_1 will reject forming the link to G_3. Thus the final coalitions are $\mathcal{S} = \{S_1, S_2\}$, where $S_1 = \{G_1, G_2\}$ and $S_2 = \{G_3\}$. This example has another potential coalition set that is discussed in the following.

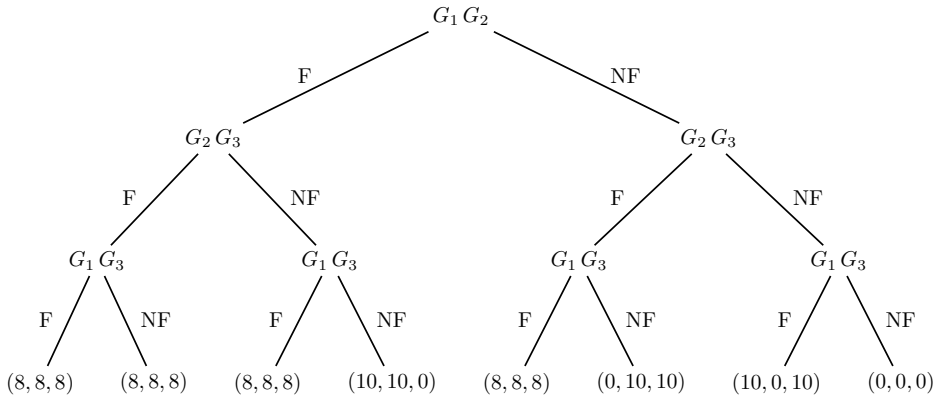

FIGURE 2. A coalition formation game among three grid service providers G_1, G_2 and G_3. F signifies that "the link is formed", while NF signifies that "the link is not formed".

The service providers are assumed to be *farsighted* when deciding which coalition to form. A GSP decides on a link not on the present level of profit but on the profit of the coalition when formation completes. The formation can be modeled as an extensive game in which each "turn" is given by the order of rule. Solving for subgame perfection by backward induction determines what the final coalitions will appear as. The tree in Figure 2 represents the coalition formation game with $v(S)$ as given in (4.1) and the order of rule as $\{G_1\,G_2, G_2\,G_3, G_1\,G_3\}$. In the figure F signifies "the link is formed" and NF signifies "the link is not formed". The outcomes are represented as tuples, where the i-th position is G_i's profit. The actions represent forming a link between pairs of GSPs. During a turn, the involved GSPs decide whether to form the link. Both GSPs must agree to the link to have it formed; if either GSP refuses, the link is not formed. There are four subgames with the action of forming a link between G_3 and G_1. Beginning with the left most subgame, forming or not forming the link results in the same outcome of $(8, 8, 8)$. So, the link may or may not be formed. In the second subgame from the left, G_1 will reject building a link and thus earn a better outcome $((10, 10, 0))$ than if the link was formed $((8, 8, 8))$. In the the third subgame, G_3 will reject the link request as there already exists a link between it and G_2. In the last subgame, both G_1 and G_3 agree to the link. Now looking at the left subgame regarding the formation of a link between G_2 and G_3, G_2 will choose not to form the link as it leads to the better outcome of $(10, 10, 0)$. In the right subgame, G_2 and G_3 agree to the link as it results in outcome $(0, 10, 10)$. But if the link is rejected, G_3 can still make the same profit by linking with G_1. In the topmost subgame located at the root of the tree, G_1 and G_2 have the option of forming a link. If the link is not formed, G_1 can link with G_3, or G_2 will link with G_3. There are six strategy profiles for the game, leading to two coalition sets of $\{\{G_1\,G_2\}, \{G_3\}\}$ and $\{\{G_1\,G_3\}, \{G_2\}\}$.

Up to this point we have assumed that computing an optimal mapping π_S in terms of cost is tractable. It is known, though, that the problem is NP-hard for $m \geq 2$ [31]. To determine the optimal coalition, a farsighted service provider would be required to evaluate an optimal mapping for all 2^m coalitions. This, by its very nature, would require exponential time. When a program is released, a substantial amount of time would be spent just on choosing a coalition, which certainly reduces the probability of successfully completing the program before its deadline. Traditionally, most research focused on minimizing the makespan of the program. The heuristic *Longest Process Time* (LPT) in which the tasks are ordered by non-increasing execution time and then scheduled using list scheduling has been shown to be a good approximation of the makespan in both identical [32] and uniform machine environments [33]. For unrelated machine environments, [34] provides an algorithm that is characterized by polynomial time complexity. Deadline scheduling heuristics such as *Earliest Deadline First* (EDF) appear to be inappropriate as they only consider the number of tasks completed before deadline and not the costs of the assignments. Deadline scheduling is a more difficult problem as it considers varying task release time and deadline, concepts that just complicate the working model.

We design a heuristic based on list scheduling that computes an approximate solution for the lowest cost mapping given a coalition S. The heuristic assumes that minimizing the total cost results in maximizing the service provider's profit. Service providers can use the following algorithm, MinCost, to compute a mapping for any coalition S that approximately minimizes cost.

Algorithm 1 (MinCost)**.**

Input: program $\mathcal{T} = \{T_1, T_2, \ldots, T_n\}$;
coalition $S = \{G_i, \ldots, G_j, \ldots, G_k\}$;
cost function $c(T, G)$;
time function $t(T, G)$;
deadline d;
payment P
Output: Map π_S
1. **for** $i = 1, 2, \ldots, n$ **do**
2. Create a priority queue TQ_i ordered by non-increasing cost for tuples $(T_i, G_j, c(T_i, G_j))$ for any i and j
3. Create an empty priority queue Q ordered by non-increasing cost for tuples $(T_i, G_i, c(T_i, G_i))$
4. **for** $i = 1, 2, \ldots, n$ **do**
5. Q.enqueue $\leftarrow TQ_i$.dequeue
6. **for** $i = 1, 2, \ldots, n$ **do**
7. $d_i \leftarrow 0$
8. **while not** Q.empty **do**
9. $(T_i, G_j, c\,(T_i, G_j)) \leftarrow Q$.dequeue

```
10.     if d_i + t(T_i, G_j) ≤ d then
11.         π.assign(T_i) ← G_j
12.     else if TQ_i.empty then
13.         π_GAP ← GAP(T, S, c, t, d, P)
14.         if the cost of π_GAP exceeds P
            or the makespan of π_GAP exceeds d then
15.             return NOT FEASIBLE
16.         else
17.             return π_GAP
18.     else
19.         Q.enqueue ← TQ_i.dequeue
20. return π
```

In MinCost, the fully polynomial time (FPTAS) GAP [34] computes an $(1+\epsilon)$-approximate map, where $0 < \epsilon < 1$, for the GSPs in coalition S. If we give it P as the cost bound and d as the time bound, it computes a mapping π_S for coalition S that approximately satisfies the following constraints:

$$C(\mathcal{T}, S) \leq P \tag{4.2}$$

$$E(\mathcal{T}, S) \leq d\,. \tag{4.3}$$

The resulting mapping will be within $(1+\epsilon)$ of both the deadline and payment.

MinCost uses a simple heuristic to find a mapping π_S with the lowest cost. The heuristic functions as follows: of all the remaining tasks without assignment, task $T \in \mathcal{T}$ is assigned to $G \in S$ if it is the lowest cost assignment and only if the assignment does not result in G exceeding the deadline. For each task T_i, for $i = 1, 2, \ldots, n$, there is a priority queue TQ_i that orders tuples $(T_i, G_j, c(T_i, G_j))$, for any i and j, by non-decreasing costs. The priority queue Q contains one tuple for each task ordered by non-decreasing cost. If the lowest cost task T_i in Q does does not exceed the deadline for service provider G_j, then T_i is assigned to G_j. Otherwise, the tuple is discarded and then the next lowest cost tuple in TQ_i is dequeued and immediately inserted in Q. If we are unable to determine a mapping by using the list scheduling heuristic, we fallback to the GAP algorithm. GAP always obtains a feasible approximate mapping if such a mapping exists. All the priority queue operations require worst-case time $\Theta(|T||S| \log |S|)$.

A mapping π_S as determined by MinCost may have service providers without any tasks assigned to them. These GSPs are safely removed from the coalition as doing so does not require modifying the mapping or costs. The only impact is increased profits for the active members of the coalition.

Even if a task assignment heuristic is found that takes constant time, coalition formation would still require an exponential amount of time due to the determination of the mappings for all possible coalitions and the necessary backward induction.

We present an example of coalition formation for a five-task program and three GSPs. The order of rule for this example is $\{G_1\,G_2, G_2\,G_3, G_1\,G_3\}$. The

TABLE 1. Costs for program $\mathcal{T} = \{T_1, T_2, \ldots, T_5\}$ and GSPs $\mathcal{G} = \{G_1, G_2, G_3\}$.

	G_1	G_2	G_3
T_1	3	2	5
T_2	3	2	1
T_3	3	5	3
T_4	2	5	3
T_5	2	5	4

TABLE 2. Execution times for program $\mathcal{T} = \{T_1, T_2, \ldots, T_5\}$ and GSPs $\mathcal{G} = \{G_1, G_2, G_3\}$.

	G_1	G_2	G_3
T_1	2	4	5
T_2	2	2	5
T_3	4	4	5
T_4	4	2	3
T_5	4	2	3

costs and execution times are presented in Table 1 and Table 2, respectively. We provide examples to further elucidate the information contained in the tables. From Table 1, GSP G_2 incurs 5 units of cost if it executes either T_3, T_4, or T_5 and it incurs 2 units of cost for executing either T_1 or T_2. If G_2 executes the entire program, it incurs cost $2 + 2 + 5 + 5 + 5 = 19$. GSP G_2 executes T_1 in 4 units of times as can be seen from Table 2. If G_2 executes the entire program, then the program completes in time $4 + 2 + 4 + 2 + 2 = 14$. Assume that the user has specified a deadline $d = 10$ and a payment $P = 20$. Each GSP computes a mapping and a cost for the mapping for each coalition using MinCost. The mappings for each coalition are given in Table 3. The profit per member for each coalition is presented in Table 4. Four strategies are the result of backward induction, but all strategies lead to the same outcome of coalitions $S_1 = \{G_1, G_3\}$ and $S_2 = \{G_2\}$. Service provider G_2 will not offer its resources as it cannot execute the program before its deadline. Therefore, S_1 will execute the program.

It is likely that GSPs would have differing heuristics when determining coalitions. Some of the heuristics would produce better solutions than the one we offered. Before a link is formed, a party could convince the other by disclosing the structure of a coalition and its associated mapping. This process guides the coalition formation. The sharing of information complicates the formation process though as the GSPs have differing ideas of the preferred coalitions and the solu-

TABLE 3. The mappings as determined by MinCost for each coalition.

S	Mapping
$\{G_1\}$	NOT FEASIBLE
$\{G_2\}$	NOT FEASIBLE
$\{G_3\}$	NOT FEASIBLE
$\{G_1, G_2\}$	$T_4, T_5 \to G_1$; $T_1, T_2, T_3 \to G_2$
$\{G_1, G_3\}$	$T_1, T_4, T_5 \to G_1$; $T_2, T_3 \to G_3$
$\{G_2, G_3\}$	$T_1, T_4, T_5 \to G_2$; $T_2, T_3 \to G_3$
$\{G_1, G_2, G_3\}$	$T_4, T_5 \to G_1$; $T_1 \to G_2$; $T_2, T_3 \to G_3$

TABLE 4. Profit for each GSP in a coalition.

S	Profit
$\{G_1\}$	0
$\{G_2\}$	0
$\{G_3\}$	0
$\{G_1, G_2\}$	3.5
$\{G_1, G_3\}$	4.5
$\{G_2, G_3\}$	2
$\{G_1, G_2, G_3\}$	3.33

tions given by backward induction may be inaccurate. But this does not hinder coalition formation.

The result of coalition formation is a set $\mathcal{S} = \{S_1, S_2, \ldots, S_k\}$ of coalitions. Any of the coalitions in $\mathcal{S}$ are able to finish the program by the deadline d and within the given cost constraint. Thus, each of them can potentially execute the program. To resolve this issue, we randomly choose one to execute the program. If we base the decision on further criteria (e.g., minimizing makespan, lowest cost), the GSPs would solve a program similarly to (3.5) but including additional objectives or constraints further imposed by the criteria.

5. Virtual organization formation framework

In this section we show how the coalition formation model presented above is used to form VOs in grid computing systems. Grid computing systems comprise geographically distributed machines, controlled and operated by independent, autonomous GSPs. We assume that these organizations exhibit welfare maximizing

behavior. We propose a framework that brings together users and service providers and supports the VO formation process in order to execute user programs.

We are proposing a resource management system, the Virtual Organization Formation Manager (VOFM), to support coalition formation among GSPs. The VOFM is implementable as a middleware component and it would be supported by the grid implementation. Instances of the VOFM would be replicated across a grid system allowing users and GSPs to interact with a local copy.

The architecture of the VOFM is presented in Figure 3. VOFM manages several lists. VO formation begins by service providers notifying the VOFM of their readiness to perform work. The VOFM records the GSPs on the *GSP ready list*, $\mathcal{G} = \{G_1, G_2, \ldots, G_m\}$. A user submits her program, $\mathcal{T} = \{T_1, T_2, \ldots, T_n\}$, and the associated program decomposition and execution profiles to the VOFM. Additionally, the user submits the amount she will pay, P, if the program is executed to completion before the deadline, d. The VOFM records $\mathcal{T}$, P, d and profiles on the *program pending list*, and notifies the GSPs in $\mathcal{G}$ that a program is pending. Each service provider G_j, for $j = 1, 2, \ldots, m$, retrieves the program details and estimates the costs $c(T_1, G_j), c(T_2, G_j), \ldots, c(T_n, G_j)$ and execution times $t(T_1, G_j), t(T_2, G_j), \ldots, t(T_n, G_j)$. These values are reported to the VOFM who records them on the *confirmed list* for program $\mathcal{T}$. A GSP need not estimate a cost or time for each task and a GSP may choose not to submit any values. Denote by $\mathcal{G}' = \{G'_1, G'_2, \ldots, G'_k\}$ the set of GSPs that have committed. VO formation proceeds among the GSPs in $\mathcal{G}'$.

In order for a GSP G'_h, for $h = 1, 2, \ldots, k$, to choose its preferred VO, it must determine the value of each of the 2^k possible coalitions. Determining the value requires the computation of a mapping. As we discussed earlier, these computations will require an exponential amount of time, and thus significantly reduce the available time for program execution. Additionally, these computations cost money that may not be necessarily recovered. Therefore, the service providers will be induced to employ heuristics. Good heuristics would increase profits and give GSPs economic advantages over their peers. An example of a heuristic that can be employed by GSPs is MinCost which was proposed in Section 4.. Consequently, it would be unlikely that the GSPs would share algorithms. But it is to the advantage of a GSP to disclose its preferred VO (coalition) and its associated mapping. This permits the other members of the GSP's preferred VO to evaluate options and if welfare maximizing, come to the same conclusion. Fortunately, a given coalition and mapping are easily verified in linear time by other parties.

There are few strategies for a GSP to determine its preferred VO in a limited amount of time. A GSP may compute its preferred VO by examining all possible VOs consisting of fewer than k GSPs. Another strategy is to randomly generate VOs in which it is a member.

Each service provider transmits its preferred VO and mapping to the VOFM, which dispenses the information to the others. For large k, the number of coalitions with known values will be minute when compared against the total number of

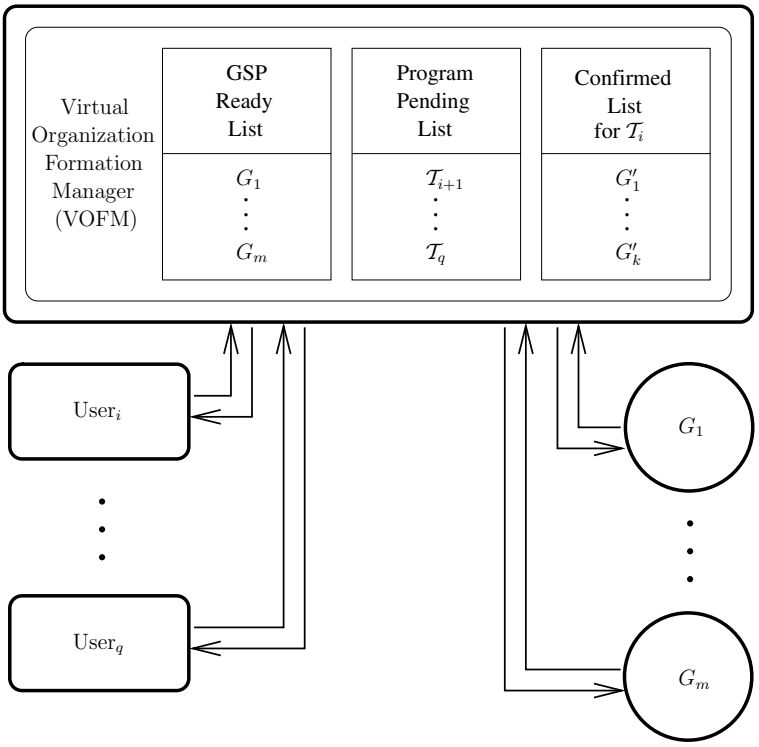

FIGURE 3. The architecture of the Virtual Organization Formation Manager (VOFM).

possible candidates. This is problematic as coalitional game theory at the very minimum assumes that the coalition values are computable. Thus, we must assign a value to these no-value coalitions. If the GSP's preferences are modeled as *risk averse* (prefers the outcomes with payoffs that are certain) or *risk neutral* (prefers the expected payoff), it seems natural that they would prefer the known coalitions to the unknown and thus, give the unknown coalitions a value of zero.

The VOFM announces a randomly generated order of rule. At this point, the link formation commences. The GSPs, being rational, perform backward induction and determine the resulting strategy profiles. One by one, the VOFM iterates through the order of rule, providing the state of coalition formation to the link associated GSPs. It then awaits responses from each GSP. As we mentioned earlier, a link is formed only when both GSPs agree; otherwise, the link is not formed. After finishing, the VOFM knows the VO set (coalition set). The VOFM now needs to select the VO that will execute the program. All VOs with no value are removed from consideration. Out of the remaining VOs, the VOFM chooses one

randomly. The selected VO executes the program. In the following we present the VO formation protocol which is employed in the VO formation process.

Protocol 1 (VO Formation)**.**

Phase I: Initialization

1. Grid service providers, $\mathcal{G}$, register their readiness with the VOFM.
2. The user submits her program, $\mathcal{T}$, to the VOFM.
3. VOFM notifies the GSPs of a pending program.

Phase II: Commitment

1. A GSP G estimates the cost and time for the pending program $\mathcal{T}$.
2. GSP G transmits costs, $c(T_1, G), \ldots, c(T_n, G)$, and time, $t(T_1, G), \ldots, c(T_n, G)$, to the VOFM.

Phase III: VO Formation

1. The VOFM dispenses the costs and times to the committed GSPs, $\mathcal{G}'$.
2. The VOFM announces a randomly generated rule of order.
3. The GSPs perform backward induction, computing the strategy profiles.
4. The VOFM iterates through the order of rule. At each link, the VOFM gives the current state of the VO formation to the link-associated GSPs. The GSPs then announce an accept or reject. If both GSPs accept, the link is formed; otherwise, the link is not formed.
5. The VO set is determined.
6. The VOFM removes from consideration all VOs with zero value.
7. The VOFM randomly selects one of the remaining VO, S.
8. VO S executes the program.

The VOFM facilitates the execution of this protocol by maintaining the required data structures. Additionally, it provides a "meeting place" as it is unlikely that all GSPs would communicate directly with one another.

6. Conclusion

In this article we proposed a model for Virtual Organization (VO) formation among autonomous Grid Service Providers (GSPs). We model VO formation using Myerson's cooperation structures. A GSP determines the VO that it will form based on its profit. Additionally, the GSPs are assumed to be farsighted. They employ backward induction to investigate what the final coalitions (VOs) will appear as. Backward induction requires computing the value of all possible coalitions, which in itself, requires determining a task mapping. This process requires an exponential amount of time, thus GSPs will use heuristics to determine mappings and coalitions. The heuristics would be considered private to a given GSP, but

fortunately, a mapping can be verified in linear time by the other GSPs. This allows them to decide in feasible time on whether to join or not the coalition. Using the above model, we proposed a resource management system, the Virtual Organization Formation Manager (VOFM), that supports VO formation among GSPs. VOFM is implemented as a middleware component with support from the grid implementation.

For future work, we plan to investigate by simulation the VO formation framework presented in this paper. We are interested in comparing the distributed approach presented in this paper to an approach that makes centralized scheduling decisions. In the framework, we assume that providers specify their costs truthfully. We would like to examine ways to extend the model in which providers misreport their costs. Finally, we plan to build a VOFM prototype.

References

[1] I. Foster and C. Kesselman, Eds., *The Grid 2: Blueprint for a New Computing Infrastructure*, 2nd ed., Morgan Kaufmann, 2003.

[2] R. B. Myerson, "Graphs and cooperation in games," *Mathematics of Operation Research*, vol. 2, pp. 225–229, 1977.

[3] I. Foster, C. Kesselman, and S. Tuecke, "The anatomy of the grid: Enabling scalable virtual organizations," *Int. J. Supercomputer Applications*, vol. 15, no. 3, pp. 200–222, 2001.

[4] I. Foster, N. R. Jennings, and C. Kesselman, "Brain meets brawn: Why grid and agents need each other," in *Proc. of the 3rd International Joint Conference on Autonomous Agents and Multiagent Systems* (*AAMAS'04*), 2004, pp. 8–15.

[5] T. J. Norman, A. Preece, S. Chalmers, N. R. Jennings, M. Luck, V. D. Dang, T. D. Nguyen, V. Deora, J. Shao, A. Gray, and N. Fiddian, "CONOISE: Agent-based formation of virtual organizations," *Knowledge-Based Syst.*, vol. 17, pp. 103–111, 2004.

[6] J. Patel, W. T. L. Teacy, N. R. Jennings, M. Luck, S. Chalmers, N. Oren, T. J. Norman, A. Preece, P. M. D. Gray, G. Shercliff, P. J. Stockreisser, J. Shao, W. A. Gray, N. J. Fiddian, and S. Thompson, "Agent-based virtual organisations for the grid," *Multiagent Grid Syst.*, vol. 1, no. 4, pp. 237–249, 2005.

[7] A. Akram and R. Allan, "Organization of grid resources in communities," in *Proc. of the 4th international workshop on Middleware for grid computing* (*MGC'06*), 2006, p. 20.

[8] G. Owen, *Game Theory*, 3rd ed., New York, NY, USA: Academic Press, 1995.

[9] R. J. Aumann and J. H. Dréze, "Cooperative games with coalition structures," *Int. J. Game Theory*, vol. 3, no. 4, pp. 217–237, 1974.

[10] R. B. Myerson, "Conference structures and fair allocation rules," *Int. J. Game Theory*, vol. 9, no. 3, pp. 169–182, Nov. 1978.

[11] M. Klusch and O. Shehory, "Coalition formation among rational information agents," in *Proc. of the 7th European Workshop on Modelling Autonomous Agents in the Multi-Agent World* (*MAAMAW'96*), Jan. 1996, pp. 204–217.

[12] I. Müller, R. Kowalczyk, and P. Braun, "Towards agent-based coalition formation for service composition," in *Proc. of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology* (*IAT'06*), Dec. 2006, pp. 73–80.

[13] O. Shehory and S. Kraus, "Task allocation via coalition formation among autonomous agents," in *Proc. of the 14th International Joint Conference on Artificial Intelligence* (*IJCAI'94*), 1995, pp. 655–661.

[14] H. C. Lau and L. Zhang, "Task allocation via multi-agent coalition formation: taxonomy, algorithms, and complexity," in *Proc. of the 15th IEEE International Conference on Tools with Artificial Intelligence* (*IACTAI'03*), 2003, pp. 346–350.

[15] L. He and T. R. Ioerger, "Forming resource-sharing coalitions: a distributed resource allocation mechanism for self-interested agents in computational grids," in *Proc. of the 2005 ACM Symposium on Applied Computing* (*SAC'05*), 2005, pp. 84–91.

[16] H.-J. Zhang, Q.-H. Li, and Y.-L. Ruan, "Resource co-allocation via agent-based coalition formation in computational grids," in *Proc. of the 2nd International Conference on Machine Learning and Cybernetics* (*ICMLC'03*), 2003, pp. 1936–1940.

[17] H.-H. Lee and C.-H. Chen, "Multi-agent coalition formation for long-term task or mobile network," in *Proc. of the International Conference on Computational Intelligence for Modelling, Control, and Automation* (*CIMCA'06*) *and and International Conference on Intelligent Agents Web Technologies and International Commerce* (*IAWTIC'06*), 2006, pp. 52–57.

[18] T. E. Carroll and D. Grosu, "A strategyproof mechanism for scheduling divisible loads in linear networks," in *Proc. of the 21st IEEE International Parallel and Distributed Processing Symposium* (*IPDPS'07*), Mar. 2007.

[19] N. Nisan and A. Ronen, "Algorithmic mechanism design," *Games and Economic Behaviour*, vol. 35, no. 1/2, pp. 166–196, Apr. 2001.

[20] D. Abramson, R. Buyya, and J. Giddy, "A computational economy for grid computing and its implementation in the nimrod-G resource broker," *Future Gener. Comput. Syst.*, vol. 18, no. 8, pp. 1061–1074, Oct. 2002.

[21] K. M. Chao, R. Anane, J. H. Chen, and R. Gatward, "Negotiating agents in a market-oriented grid," in *Proc. of the 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid* (*CCGRID'02*), 2002, p. 436.

[22] J. Gomoluch and M. Schroeder, "Market-based resource allocation for grid computing: A model and simulation," in *Proc. of the 1st International Workshop on Middleware for Grid Computing* (*MGC'03*), 2003, pp. 211–218.

[23] R. Wolski, J. S. Plank, J. Brevik, and T. Bryan, "Analyzing market-based resource allocation strategies for the computational grid," *International Journal of High Performance Computing Applications*, vol. 15, no. 3, pp. 258–281, Aug. 2001.

[24] D. Grosu and A. Das, "Auctioning resources in grids: model and protocols," *Concurr. Comput.: Pract. Exper.*, vol. 18, no. 15, pp. 1909–1927, 2006.

[25] D. Grosu, "AGORA: an architecture for strategyproof computing in grids," in *Proc. of the 3rd International Symposium on Parallel and Distributed Computing* (*ISPDC'04*), July 2004, pp. 217–224.

[26] R. J. Aumann and R. B. Myerson, "Endogenous formation of links between players and of coalitions: an application of the shapley value," in *The Shapley Value: essays in honor of Lloyd S. Shapley*, A. E. Roth, Ed., Cambridge University Press, 1988, pp. 175–191.

[27] J. Yang, A. Khokhar, S. Sheikh, and A. Ghafoor, "Estimating execution time for parallel tasks in heterogeneous processing (HP) environment," in *Proc. of the Heterogeneous Computing Workshop*, 1994, pp. 23–28.

[28] M. A. Iverson, F. Özgüner, and L. Potter, "Statistical prediction of task execution times through analytic benchmarking for scheduling in a heterogeneous environment," *IEEE Trans. Comput.*, vol. 48, no. 12, pp. 1374–1379, Dec. 1999.

[29] L. S. Shapley, "On balanced sets and cores," *Naval Research Logistics Quarterly*, vol. 14, pp. 1589–1594, 1967.

[30] L. S. Shapley, "Cores and convex games," *International J. of Game Theory*, vol. 1, pp. 1–26, 1971.

[31] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., 1990.

[32] R. L. Graham, "Bounds on multiprocessing timing anomalies," *SIAM J. Applied Math.*, vol. 17, no. 2, pp. 416–429, Mar. 1969.

[33] D. K. Friesen, "Tighter bounds for LPT scheduling on uniform processors," *SIAM J. Comput.*, vol. 16, no. 3, pp. 554–560, 1987.

[34] P. Efraimidis and P. G. Spirakis, "Approximation schemes for scheduling and covering on unrelated machines," *Theory Comput. Sci.*, vol. 359, no. 1–3, pp. 400–417, Aug. 2006.

Thomas E. Carroll
Department of Computer Science
Wayne State University
5143 Cass Avenue
Detroit, MI 48202
USA
e-mail: `tec@cs.wayne.edu`

Daniel Grosu
Department of Computer Science
Wayne State University
5143 Cass Avenue
Detroit, MI 48202
USA
e-mail: `dgrosu@cs.wayne.edu`

Economic Models and Algorithms for Distributed Systems, 83–104
Book Series: Autonomic Systems

Towards Dynamic Authentication in the Grid – Secure and Mobile Business Workflows Using GSet

Jürgen Mangler, Erich Schikuta, Christoph Witzany, Oliver Jorns, Irfan Ul Haq and Helmut Wanek

Abstract. Until now, the research community mainly focused on the technical aspects of Grid computing and neglected commercial issues. However, recently the community tends to accept that the success of the Grid is crucially based on commercial exploitation. In our vision Foster's and Kesselman's statement "The Grid is all about sharing." has to be extended by "... and making money out of it!". To allow for the realization of this vision the trustworthyness of the underlying technology needs to be ensured. This can be achieved by the use of gSET (Gridified Secure Electronic Transaction) as a basic technology for trust management and secure accounting in the presented Grid based workflow. We present a framework, conceptually and technically, from the area of the Mobile-Grid, which justifies the Grid infrastructure as a viable platform to enable commercially successful business workflows.

Mathematics Subject Classification (2000). Primary 91A40, 68M14, 68T99; Secondary 91A10.

Keywords. Grid computing, virtual organization, self organization, cooperative game theory

1. Introduction

Nowadays it can be learned from the developments in the IT market that there is a shift in the source of profits, from sales of products to the provision of on-demand services [1]. This leads to new business models, building on application and resource sharing. Thus for companies to be successful in the IT market the availability of techniques allowing for integration and collaboration of resources of any kind is a crucial issue.

The Grid promises the revolution of the Internet by a novel and advanced support for collaboration providing homogeneous access to virtualized resources without revealing the heterogeneous manner of the underlying real world. Basically the Grid resembles a distributed computing model supporting the selection, sharing, and aggregation of geographically distributed "autonomous" resources dynamically at runtime depending on their availability, capabilities, performance, costs, and users' quality-of service requirements via Grid-enabled Web Services [2].

In the Grid community many people have different ideas about what a Grid really is. This motivated us to define necessary characteristics to allow for business workflows in the Grid infrastructure, which are based on the findings of the successfully completed research project "BIG – Business in the Grid" [3] Weishaeupl et al. The specific characteristics for a *Business Grid* according to our definition are:

- *Transparency*: The parties involved in sharing of resources are anonymous to each other. That means the consumer of a resource does not need any knowledge about the provider of the resource and vice versa.
- *Generalized Quality of Service*: QoS is more than technical properties. It is better described by the notion of SLA (Service Level Agreement) and has to comprise all necessary aspects of business resulting in a "trust relationship" between customer and provider. This can be confidentiality, data integrity, non-repudiation, accountability, ...
- *Brokerage*: Resources are made available by a Broker, which is in our terminology a policy-based mediator of services and business workflows aiming for maximizing profit of all partners.

In our vision it will be possible to sell software and resources as a service and not as a good in the near future. For example, "storing photos and distributing them among friends" can be as simple as using a telephone. All what is necessary is a simple interface to the services on the Grid. This includes interfaces for both, the imaging functionality (format conversions, image processing) and the necessary physical resources (processor cycles and storage space).

In this chapter we present a framework, conceptually and technically, from the area of the Mobile-Grid, which justifies the Grid infrastructure as a viable platform to enable commercially successful business workflows. Trust is implemented by the use of the gSET (Gridified Secure Electronic Transaction) model [4] as basic technology for trust management and secure accounting.

The trust and credential management in a commercial context has to handle hidden (covered) but trustable information. We identified two crucial ingredients for successful business workflows:

- The service requester wants to provide a minimal set of information to a potential business partner. Private information like credit rating or payment details (credit card numbers) should not be disclosed.

- The business partner has to establish a trust relation to the service requester in order to accept a business.

Secure Electronic Transaction (SET) was developed for this concern and is well known in the classic e-business area. The *dual signature* is the key mechanism inside SET. gSET adapts the concept of SET, namely the dual signature, for virtual organizations (Grids). Trust management in a commercial, business environment (e.g. for Application Service Providers) is needed to manage confident service requester information, as accounting data and resource usage data.

Existing trust management solutions, like the ones discussed in the related work Section below, do not fulfill the two key requirements of trust and requester privacy. There are no already existing solutions yet, and even combining available methods provides no satisfying remedy for this concern.

Mobile clients are omnipresent, but in order to utilize them not only as vendor-locked devices with a fixed set of services we need to establish an additional model with the following characteristics:

- Everybody can be consumer and provider.
- A virtual organization can be built up anywhere, anytime by anybody, it should be not dependent on a mobile service provider.
- To be part of the market, no expensive, local resources (such as hardware or software) are necessary, everything that is needed is a simple, easy to use and affordable access to services via the Grid, which is exemplified by a mobile device in this chapter.

We think that providing a secure way of handling the payment aspect in this new way of doing business is crucial to its success.

Our contribution is structured as follows. First, we describe the related work for trust management and accounting and underline the differences to gSET. The we introduce the big project (Section 3.) which provided the theoretical foundations for the presented approach and discuss the requirements of dynamic authorization (Section 4.) for enabling novel business models in the Grid. Section 5. explains the concept of SET with the dual signature mechanism and describes its implementation on the Grid - gSET. The business model and corresponding scenarios are described in Section 6.. The service model based on gSET and its general protocol that defines the message flow/workflow in such a market are explained in detail in Section 7.. In Section 8. we will have a look at the performance characteristics of our implementation on a mobile client. We will conclude with possible extensions and future work.

2. State of the art

There are more than two billion mobile subscribers [5]. Mobile devices are rapidly increasing in number and the applications running on these devices are gaining

in richness and complexity. Despite of all the efforts in terms of improving the communication technologies and the performance of the mobile devices, there have been some inherent constraints within the architecture of these devices that strictly confine the applications running on such devices. Constraints include in fact the essential resources like computational power, battery life, and memory. The Grid community has realized the importance of mobile devices and there have been many efforts [6–11] to make these devices part of the Grid infrastructure. Thus, the static notion of the Grid is involving into what has been termed the "Mobile-Grid".

In a Mobile-Grid environment, any of the resources or clients can be mobile. This implies different possibilities for resource-client communication such as Mobile-Mobile, Static-Mobile and Mobile-Static. The task of the broker becomes more challenging with reference to resource selection and client communication. Some resources are no longer usable while others become available. Mobility patterns of the resources [12] can play a very important role for their selection.

Preparing Grid for the business is a challenging task but still the Grid community finds enough motivation [4] to invent means of doing business in this novel infrastructure. The Economy Grid [13] describes a hierarchy of involvement of business in the Grid as a four layer model. These layers are from the bottom up: Grid using Economic Principles, Selling Grid Software, Business enabled Grid, and Business models on Grid. To enable Grid for any successful business, the key issues that need to be evolved include management of trust, authorization, privacy of accounting and payment.

Basic security services in Grids are provided by PKI (Public Key Infrastructure). The quality of PKIs depends on the management of the certificates and the education of the users. Certification Authorities (CAs) need to verify the identity of all participants (users, resources, and services). Many bureaucratic and technical challenges are related to issuing a certificate (photo ID of users, etc.). The organizational effort is high. By PKI a strong authentication mechanism is provided.

Nevertheless, the authorization of users, the insight of who consumes which specific services is not solved by the user's authentication alone. Dynamic, ad-hoc authorization is not possible.

A first approach to manage authorization assertions was done by GSI [14] with "gridmap" files. It utilizes directly the user certificate identifiers and maps them to local user accounts. The scalability of the gridmap file approach is limited. The trust management by gSET provides account management without organizational overhead for the Service Provider.

The Community Authorization Service (CAS) [14, 15] allows to express policies regarding resources distributed across a number of sites. Similarly, the Virtual Organization Membership Service (VOMS) [16] also gives the capability to provide authorization information by a secure server that the local site has chosen to trust. The three different services have no dependencies among each other. At

GGF13, OGSA-WG [17, p. 18] underlined that, by now, no real solution for VO management exists.

GridBank [18] provides services for accounting. Our gSET approach can integrate this capability and also supports other accounting modules, by the Account Provider (AP), explained in detail below. No wrapper was built until now for GridBank, because gSET is already implemented with WSRF.

SweGrid Grid Accounting System (SGAS) [19] can also be integrated and gSET provides an accounting enabled third party authorization service.

3. Requirements analysis – The BIG project

From 2004 to 2007 the Business In the Grid (BIG) project took place and was driven by the following goals: Firstly, make business aware of Grid technology and, secondly, try to explore new business models. We disseminated Grid computing by mainly concentrating on the central European market and interviewed several companies, with the purpose to analyze the IT market in order to find out if there is a market potential to use Grid technologies in a commercial environment. In detail, the project had the following goals, as described in [3]:

1. **Revisiting of existing (E-)business models for the Grid.** Existing (E-)business models were revisited for a possible adaptation to the Grid. Reasons for the failure of existing business models for E-Commerce in the Internet and Business to Business (B2B) were analyzed. It was observed that the Grid can provide chances of success for these models by new *transparent layers* (included security mechanisms, support of dynamic services, etc.).
2. **Market potential analysis and information dissemination.** Information dissemination and market potential and demand analysis was mainly focused on the Austrian economy landscape, but some other countries such as Germany, Czech Republic and USA were also kept in view.
3. **Development of novel business models for the Grid.** The main goal of the BIG project was to find novel business models enabled by the Grid infrastructure. New ways of doing business in the Grid were discovered. It was realized that there are possibilities for dynamic collaborations, new project workflows, software on demand, dynamic resource-management, resource on demand, application service providers, and other Grid information society components.

One of the key findings of the BIG project was the requirement of dynamic authorization for the provisioning of novel business models within the Grid infrastructure.

4. The need for dynamic authorization

Trust and security are often claimed in Grid computing as some of the functional differences to earlier developments in the Web and distributed computing [20]. Beyond organizational boundaries, virtual organizations need a trustable and secure infrastructure to utilize autonomic resources and services. Security describes a field of activities to guarantee the privacy, integrity and availability of resources.

Although Grid research has gone a long way from the first steps in the 1980's, it is still far from providing plug and play tools that can easily be deployed and maintained. Current Grid Solutions are mostly custom fitted systems deployed by big players like IBM for big customers or installed at scientific research facilities fitted to the needs of scientists that are often very different from the requirements a business poses on a new technology.

Furthermore, most Grid projects are still very introspective. They are intended to share the resources of some high performance computing center or provide a common file system to members of a research group. They focus on strongly coupled rather small virtual organizations while the vast majority of the benefits Grid computing could leverage keep lying bare in the Internet.

Publicly available Grid services however will serve a heterogeneous group of customers not bound together by being part of the same business or working at the same scientific working group. This demands for tools to dynamically manage big communities of loosely coupled entities providing and consuming Grid services without excessive administrative overhead.

Another problem hindering the progress in publicly available Grid services is the method of payment. Conventional services like delivering pizza or books or booking a flight that are offered over the Web require human interaction anyway. Therefore payment can be handled interactively by the requester of the service. However Grid services typically only involve machine to machine interaction and should be processed transparently for the user. Today there are very few projects that address this problem.

One of the most promising usages of Grid computing is distributed storage and manipulation of data. For a flexible and universal usage of Grid resources, it will be necessary to abstract the process of distribution and make it transparent for the Grid user. A metaphor often used for this abstraction is the electric power Grid. What seems to the user as a simple push on a button is backed by a sophisticated framework of power plants and transformer stations that span (sometimes) the whole continent.

To obtain this abstraction it will be crucial to have an opportunity to provide a flexible way to manage the access to Grid resources anonymously as well as trusted. While the provider of a resource wants to secure that the user of his facilities is trustable, the user himself is interested in maintaining his anonymity, while gaining easy access to the services.

Staying with the metaphor of the electric power Grid, the user does not want to have to identify himself to an operator of the electricity provider if he turns

on the light (and therefore uses their services). In turn the electricity corporation does not want to face the bureaucracy arising with identification for every watt they send him. Instead they are interested in knowing him as a reliable customer with a sufficiently covered bank account to pay for his demand of electricity.

In the course of the BIG project we developed gSET [4], a mapping of Secure Electronic Transaction (SET) to the Grid environment, which can fulfill these requirements. It is intended to show a possibility to handle authorization and payment in loosely coupled Grids of commercially interacting units. Using Grid services should be as easy as paying with a credit card (or even easier). The method described provides an authorization mechanism that is *request based* and *dynamic*. *Request based* means that the authorization to use a service is evaluated at the time the service is to be invoked and can depend also on the contents of the service invocation not only on the role and rights of the service requester. Furthermore in gSET the authorization to use a service depends on the service requesters credit at a configurable account provider, making the solution highly *dynamic* and easy to integrate into existing accounting systems. Due to this delegation a gSET enabled service does not require an extensive account management by the service provider and is therefore perfectly suitable for the commercial service provision.

5. Gridified Secure Electronic Transaction (gSET)

The SET [21–23] protocol enables secure credit card transactions over the Internet and allows the secure transfer and verification of credit card information between two business partners. SET was developed by MasterCard, Visa, and others who intended to enhance privacy and security for on-line transactions. In the following we will extend the SET method to gSET, a protocol for secure transactions on the Mobile Grid. We will justify its usage by specific application scenarios.

In a SET transaction, the payment information is hidden from the merchant, but the merchant can verify the information (e.g. credit card limit) through a payment gateway trusted by both sides. Vice versa, the payment gateway (including the issuer and brand) can not read the confident order information. Nevertheless, the integrity of the whole message can be verified by both parties. The mechanism providing this functionality in SET is called *dual signature*. The dual signature separates the payment from the order information in a way that allows verifying the integrity of the data without disclosing all information and thus ensuring privacy.

To achieve this, both message parts are hashed, the hashes are concatenated and signed. Each receiver then gets his message part and the hash of the other part. By hashing his part of the message and comparing this hash with the received hash of the other message part he can then verify the integrity of the message.

SET was designed to enable secure credit card transactions on the Internet. SET could not gain commercial acceptance because back then there was no public key infrastructure on the Web, bad usability for the customers due to the involve-

ment of authentication certificates and the high CPU requirements of the method. With gSET, the proposed extension of SET to Grids, the first two problems are solved because Grids already have these two ingredients naturally available in their infrastructure. Further we will also show that current mobile devices have enough power to easily handle the gSET authentication workflow. So gSET is a very promising solution for transactions on the Mobile Grid.

5.1 gSET

We propose a business model through a state of the art financial mechanism called gSET [4]. gSET keeps the financial information separate form the order information and enables both parts of the information to reach the appropriate parties i.e. the Trust Manager and the Service Provider respectively.

With gSET it is possible for a service requester (client, consumer, card holder) to access a service (resource). The requester receives credentials depending on certain criteria. The criteria can be client credit rating, reliability, trustworthiness or other client properties. These criteria are calculated internally by an account provider, e.g. according to past business cases. When a service provider wants to establish a trust relationship with a client, it does so by asking a trust manager, which acts as a middleman. The criteria are not disclosed to the service provider.

So gSET allows a service provider to verify the trustworthiness of a client and decide, if it will trust and grant access to that client. In the other direction it is also possible for a client to ensure the delivery of the service with the agreed quality, because the trust manager has a contract to the account provider as well as the service provider, otherwise no contract can be established. The contract between the trust manager and the service provider guarantees that the service provider is a valid business partner. So QoS issues that gSET addresses are

- confidentiality,
- integrity,
- authentication and authorization and
- non-repudiation.

Therefore gSET can be classified as a higher security service [24, p. 285]. In Grids, PKIs are used to provide the basic security services (e.g. TeraGrid [25], DataGrid [26], Gridbus [2] and others).

The adoption of gSET is obvious and simple, because of the existing PKIs in Grids. The certificate management, the establishment of a reliable PKI, and its usability was one of the reasons, that SET had no commercial success in e-business. A prerequisite is a strong certificate policy with e.g. photo IDs to ensure the quality of all signature and encryption mechanisms used in gSET.

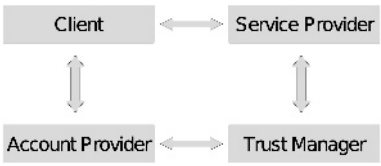

FIGURE 1. gSET architecture.

5.2 Architecture

With gSET we provide an approach to construct services, which use the SET workflow to authorize requesters dynamically. gSET provides a secure accounting mechanism.

Figure 1 shows the overall gSET architecture. Like SET itself, gSET conceptually relies on a four tier architecture. This enables a maximum distribution and redundancy for the dynamic management of trust and authorizations to services. The actors in the gSET architecture are Clients, Account Providers, Service Providers, and Trust Managers.

Clients are service requesters. In the SET concept this is originally the consumer (card holder) who is required to have an account issued by an Account Provider. The service requester may be unknown to the Service Provider before the initial request of a service.

Account Providers are like credit card companies (brand, issuer) in the original SET concept. They issue accounts (credit cards) and have a trust relation with their customers. The trust between Account Providers and Clients is based on assurances like monetary entities or organizational relations. Of course account providers may also include paypal, google checkout, etc.

Service Providers make services available. For example they provide storage space etc. In the original SET concept they were the on-line sellers (merchant). They need at least a relation to one Trust Manager as a gateway to an Account Provider. By this the Service Provider gets guarantees about the Client's behavior without harming the privacy of the Client.

Trust Managers are the link between the Service Providers and the Account Providers. They manage the payment and verify the accounts of the Clients at the Account Provider. In the original SET concept this was the payment gateway. The Trust Manager must have a contract with the Client's Account Provider to establish a successful authorization. One Trust Manager can handle the gateway requests to different Account Providers.

The authentication of all parties is done through valid and trusted certificates. Grid's PKI provides these inherently with a network of CAs and certificates for clients and service providers.

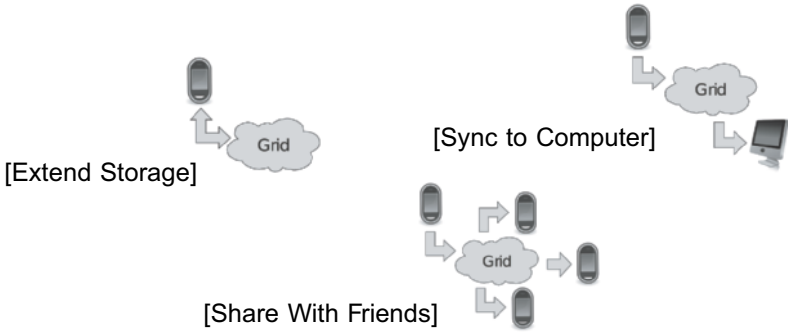

FIGURE 2. Use cases.

6. Scenario and business model

Envision the following figurative example: Jane is visiting Vienna and wants to record her memories by taking photographs of all the historical places using the 3.2 Mega Pixel camera built into her cellular phone. But after a few pictures, she realizes that her mobile phone runs out of disc space. She suddenly recalls the Grid services accessible by a broker. She moves the pictures to a certain subdirectory, which triggers a request to the broker, which returns a list of Service Providers and their rates. One of these Service Providers is selected automatically according to a set of preferences. After the successful migration of the images they are deleted from the directory and replaced by a ticket that holds the information to later on retrieve the images. Jane also wants to share the images with her friends so she distributes the ticket through a social networking site.

However this simple scenario can lead to several slightly more complex examples:

Enlarging the own storage space is the most simple case (see Figure 2). Rarely used data should be transferred to the Grid, and it should be possible to retrieve it transparently on user demand, whereas often used data can be held on the device. A ticket that allows potentially multiple accesses to the data is required.

Syncing the data with a home PC is depicted in Figure 2. In this case a ticket, which can be used to access the data once is needed. After returning, the tickets can be synced to the home computer to allow for retrieval from the grid.

Distributing data among friends is in our view the most interesting scenario (see Figure 2). We assume the security preferences in this case to be low (imagine Jane to take photos, and immediately sending tickets to all of her friends, who can then follow her sightseeing tour). In this case we imagine the process to return a certain number of tickets that allow for a single access to a certain picture.

To establish certain business models around these scenarios we have to take a closer look at when and why payment is necessary:

- In the first scenario the most suitable business case would be to demand payment for every access to the file.
- The second scenario could be extended by assuming that the user creates a large video which additionally is converted to a suitable format before downloading it to the home PC.
- In the last scenario a valid business case could be to only charge the money when the friends really access the photos.
- Another scenario could include distributing tickets to users, and charging them when they actually decide to access the content (again maybe with the service to provide different formats).

So our architecture makes it possible to sell and buy services on the Grid. Our business models are based on simple and straightforward principles.

- The customer can buy things without revealing financial information to the vendor.
- Customers and vendors both express their trust in a mediator who assures the customer about the reliability of the vendor and guarantees the vendor about the financial status of the customer.

The Trust Manager plays a very important role in a financial transaction in the sense that both the Service Provider and the Account Provider have trust in it. The Account Provider verifies the Client's financial status and thus the Service Provider has no hesitation to process the order. In the scenario that we have presented here, the customer buys storage services from a vendor.

The customer's financial credibility is maintained by the Account Provider that may be at the same time the mobile service provider and charges the customer on a regular basis. The Account Provider pays for the service on behalf of the customer and this amount is added to the monthly bill of the customer. Account providers need to have a contract with the Trust Manager.

In order to make the Service Providers discoverable, we will introduce a Broker into the picture. The Broker itself is not part of the business workflow, but acts as a portal to make Service Providers discoverable. He is intended to be a policy-based mediator aiming for maximizing profit of all partners. The client discovers suitable service providers by sending a set of preferences to the Broker which will respond with a list of suitable service providers. On the other hand can the Broker also act as a ranking system. This ranking can also be done by the broker based on sales patterns, by collecting data about contracts (see Section 7.).

7. Integrating gSET with a mobile client

As mentioned before, there are several parties in this model: Client, Account Provider, Trust Manager and the Service Provider. Additionally we introduced the Broker to make Service Providers discoverable by Clients.

In our scenario the Client is the owner of a mobile device who wants to buy extra storage space. The Service Provider can be any vendor, registered with a Broker, which in turn manages, authenticates and mediates the resources in the Grid. The Account Provider can be a credit card authenticator or a mobile network operators service provider. Accordingly the customer can be billed via credit card or the monthly phone bill. The Trust Manager plays the important role of a business-deal-mediator and is considered reliable by both, the Client and the Service Provider.

7.1 Considerations regarding mobile devices

For the development of the service architecture we had to consider that almost any mobile device provides only low processing power and has severe memory constraints which renders security mechanisms and protocols that are based on e.g. public keys rather impossible to implement. Anyhow, as in our opinion security is a key to the success of the system we followed the approach to embed security aspects into the workflow which eventually even leverages the architectural design. Alternatively, security design atop existing service architecture is not only difficult to develop but in most cases turn out to be error prone and unusable. We defined our architecture based on the following security related assumptions:

- The communication channel between a Client and a Broker is insecure.
- The communication channel between a Broker and a Service Provider is insecure.
- The Client and the Broker share a secret, exchanged via Diffie-Helmann, which we further on refer to as pswd.
- If computational limitations exist, the data exchanged between Client and Service Provider can not be encrypted by the mobile device. However we have to ensure the identity of communication partners as well as the integrity of the data.

To achieve the goal of at least ensuring the identity of communication partners and for securely transmitting data fingerprints we pursue the approach of attaching pseudonyms to messages. The advantage of this approach is that the generation of the pseudonyms needs less processing time than full encryption (see Section 8.). We try to reduce the computational intensive use of cryptographic algorithms as much as possible.

In the following we give a short introduction to the pseudonym approach: We assume that the Client and the Broker share a secret key ($pswd$) as well as an initial value (rand) generated by the Diffie Hellman cryptographic protocol [27]. We also use a schema that is based on the keyed hash function HMAC [28]. The keyed hash function HMAC that is denoted as h_k can be initialized with a random value r. As shown in the following equation, we build a chain of hash transaction pseudonyms by consecutively applying the hash function. $h_k^i(r) = h_k(h_k^{i-1}(r)), i = 1, 2, \ldots$ This schema allows to use different pseudonyms for each transaction whereas

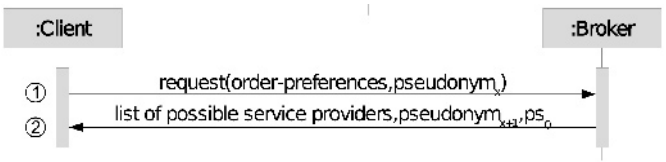

FIGURE 3. Requesting a list of storage providers.

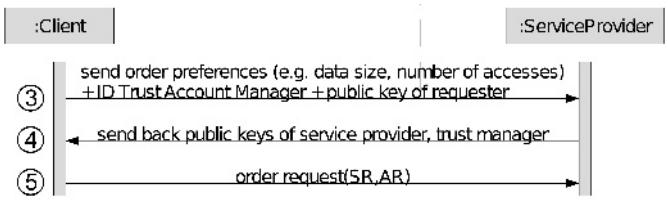

FIGURE 4. Order request.

one pseudonym cannot be linked to any other pseudonym that was used before. Furthermore, this schema makes it impossible to predict the respective next pseudonym.

7.2 Tickets

The purpose of a ticket in our system is to access data. The ticket holds the information and a pseudonym to access the data. With a ticket even clients not part of the PKI can participate as described in the scenarios. The secure distribution the tickets to other clients is the responsibility of the client.

7.3 The mobile gSET workflow

Figure 3 shows the first steps of the proposed scenario. The Client sends order preferences to the Broker. These preferences may include attributes like storage size, maximum price, additional needed services (like format conversion), etc. The $pseudonym_x$ is a random value, that is used as the base of hash transaction pseudonyms as described above. Upon receipt of these attributes the Broker sends back to the Client a list of suitable Service Providers along with $pseudonym_{x+1} = HMAC(pseudonym_x, pswd)$ and a value ps_0. The random number ps_0 is calculated by the Broker as the anchor pseudonym for further messages exchanges regarding the current transaction.

As seen in Figure 4, the Client selects one of the Service Providers from the list and forwards the order preferences together with its public key and the ID of its Account Provider. The Service Provider has to check if the order preferences

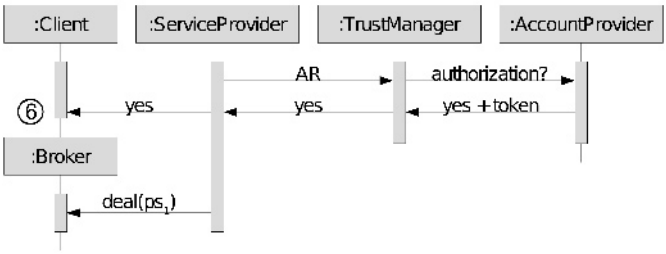

FIGURE 5. Successful negotiation.

can be met (there could be circumstances, that the Broker was not aware of when selecting the Service Provider; e.g. intermediate contracts negotiated by different Brokers), as well as if its associated Trust Manager has a relation to the Account Provider of the Client. If both prerequisites are met the Service Provider returns its own public key plus the public key of its Trust Manager, signed with the private key of the Service Provider and encrypted with the public key of the Client.

The Client then generates the two part message and the dual signature:

- SR denotes the service request part which includes the order preferences and $ps_1 = HMAC(ps_0, pswd)$ (later on used for communication between the Service Provider and the Broker) and is dedicated to the Service Provider.
- The authorization request part (AR) is dedicated to the Trust Manager and contains the credentials of the account (provided by the Account Manager) as well as the amount and currency to be authorized.
- The dual signature (DS) consists of the concatenated hashes of SR and AR and is signed with the Clients private key.

AR, the hash of SR and the dual signature are combined into one message and encrypted using hybrid AES/RSA encryption using the Trust Provider's public key. The same procedure is done with the combination of SR, the hash of AR and the dual signature using the Service Provider's public key. The Service Provider decrypts and verifies its part (SR). Upon success it forwards the second part (AR) to the Trust Manager (see Figure 5) which also verifies that no security breaches occurred. The Trust Manager forwards the authorization request to the Account Provider which after confirming the profile of the Client sends its confirmation report back. The Trust Manager sends this confirmation report along with a payment authorization token to the Service Provider. The Service Provider then informs the Client and the Broker of the successful deal. The deal confirmation for the Broker includes the ps_1, which can be used to identify the transaction (by comparing it to $HMAC(ps_0, pswd)$).

The Client then also should confirm the transaction, identified by $HMAC(ps_1, pswd)$ (see Figure 6).

FIGURE 6. Confirming deal.

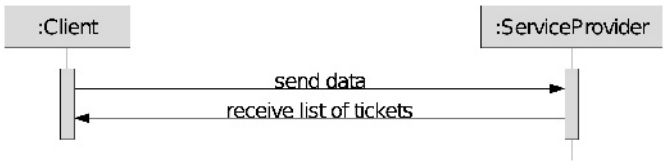

FIGURE 7. Storing data.

The deal confirmation helps the Broker to link preferences to successful deals, which could be used to improve the result quality.

Finally (see Figure 7) the data is sent to the Service Provider, either ...

- signed by the private key of the Client and encrypted by the public key of the Service Provider, or
- when CPU power and bandwith are precious, protected by pseudonyms calculated according to the proposed scheme ps_1: $fp = HMAC(hash(data), ps_1)$. This at least allows to check the identity of the communication partner and the integrity of the data.

The Service Provider then returns the tickets that are necessary to access the data, again protected by one of the above means.

8. Performance analysis

To justify the performance of the proposed gSET method for practical usage we examined two cases, firstly a comparison to an existing Grid method, the *gridmap* file authorization in GT4, and secondly the evaluation of gSET in a real Mobile Grid environment.

8.1 gSET versus gridmap

The *gridmap* method manages authorization assertions by directly utilizing the user certificate identifiers and mapping them to local user accounts. The scalability of the gridmap file approach is limited. The account administration (trust management) to authorize users is a hard task to the involved parties. The trust

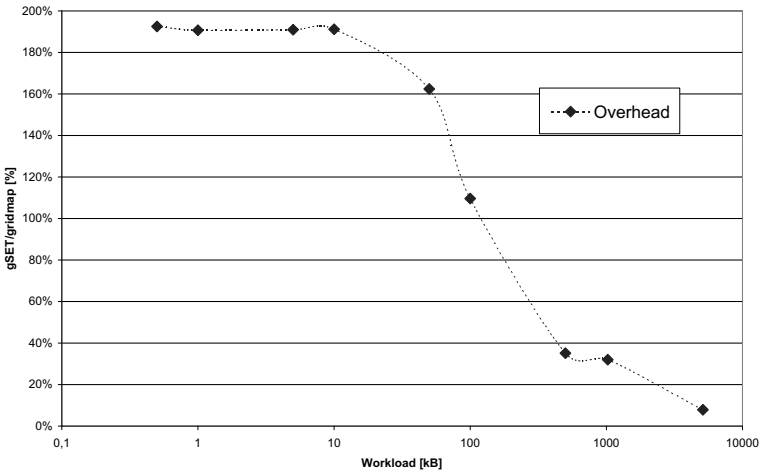

FIGURE 8. Overhead of gSET versus gridmap.

management by gSET provides account management without organizational overhead for the service provider.

We evaluated gSET by comparing the gSET enabled storage service with same storage service based only on GT4, respectively these services are called in the following *gSETstore* and *gt4store*.

The gt4store service uses gridmap file authorization and does not provide the following functions without violating the requesters privacy:

- Secure accounting
- Verification of requester credit rating
- Dynamic authorization

The services were deployed in a GT4.0.1 Java core container (with TLS) running on Debian sarge Linux (kernel 2.6.12.3), on a Dell PowerEdge 2850 with two Intel Xeon CPU 3.60GHz processors, 4GB of RAM with 1,400 GB storage. The Java clients run on a Windows XP workstation. The interconnection between server and client was a switched 100MBit Ethernet.

We measured the execution time for different workloads (data transfer size). The total execution time consists of the service execution time (authorization time, request processing), the transfer time (network latency and throughput, TLS), and the client time (construction of the request). Every time was measured 50 times and the median was used for the statistical analysis.

The differences in the total execution time between the gSETstore and the gt4store result only from the authorization time differences.

Figure 8 shows the relative overhead of gSET versus gridmap authorization. The overhead is qualified because of the functional advances of gSET. By increasing the workload the authorization overhead decreases relatively, therefore Figure 8

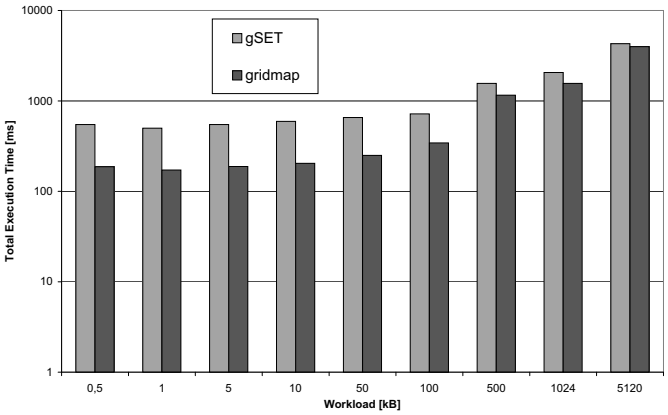

FIGURE 9. Total time of gSET and gridmap.

shows that the gSET overhead is nearly constant for small workloads. Nevertheless, the curve decreases because of the dominance of the request processing time for high workloads and the negligible authorization time.

Figure 9 shows the absolute execution times of gSETstore and gt4store calls for different workloads. It shows that the execution time for small workloads (storing 512 Byte up to 50kB) is constant. This shows that the request processing time can be neglected for calculating the real overhead of gSET for small workloads.

Summing up, gSET shows comparable performance to the gridmap method by providing additional functions guaranteeing the privacy of the requester, which are necessary for enabling business workflows in Grids.

8.2 Evaluation of gSET in a real mobile grid environment

To prove the viability of the model for mobile devices we implemented our scenario on a Neo Freerunner by Openmoko Inc. [29]. We also did a performance analysis of the used cryptographic algorithms. The device should be similar to current mobile devices like the HTC G1 or the Apple iPhone, and has the following characteristics: ARM920T processor rev 0 (v4l), 400 Mhz.

We tested runs with 1000 iterations of all cryptographic methods involved, using libssl0.9.8g-14, yielding the results collected in Table 1.

It can be noticed that the device is powerful enough CPU-wise to accomplish its task. As the device runs Linux it was easy to implement our scenario. The effort for the mobile device for executing the steps described in Section 7., Figure 3 to Figure 6 is constant, and consists of the following steps (denoted by steps 1 to 7 in respective Figures:

1. $pseudonym_x = pswd(rand)$
2. $pseudonym_{x+2} = pswd(pseudonym_x)$
3. –

Table 1

Algorithm	Key Size	Payload Size	Time (s) for 1000 Iterations	Mean
RSA	2048 bit	256 bit	9.1	0.0091
HMAC-SHA1	12 byte	256 bit	1.5	0.0015
AES-256-cbc	256 bit	1 KiB	1.5	0.0015
AES-256-cbc	256 bit	10 KiB	4.5	0.0045
AES-256-cbc	256 bit	100 KiB	40.6	0.0406
AES-256-cbc	256 bit	1000 KiB	410.7	0.4107
SHA1	x	1 KiB	0.3	0.0003
SHA1	x	10 KiB	1.1	0.0011
SHA1	x	100 KiB	7.8	0.0078
SHA1	x	1000 KiB	74.7	0.0747

4. $S = D_{RSA}(M), O = D_{AES_S}(N), (O_1) = V_{O_2}$
5. Details:
   - DS: $S((O_{SR}) + (O_{AR}))$
   - SR: $ps_1 = pswd(ps_0), E_{AES_S}((AR) + DS + O + ps_1), E_{RSA}(S)$
   - AR: $E_{AES_S}((SR) + O), E_{RSA}(S)$
6. $E_{AES_S}(yes) + E_{RSA}(S)$
7. $pswd(ps_1)$

S denotes *sign* which is always done with the own private key, V denotes $verify$ which is always done with the communication partners public key, E stands for *encrypt* (by either RSA or AES, for AES followed by the key) and D denotes *decrypt*. The letter M is used for RSA encrypted AES keys, N for AES encrypted data, O for XML payload that holds preferences or information. For the sake of simplicity we assume O to have an average size of 1 KiB.

When using the values from Table 1 this results in a constant factor of $(4*) + (6 * F_{RSA}) + (4 * F_{AES}) + (5*) = 0.0681$ seconds spent for doing the necessary work on the mobile client, which corresponds to our measurements. Of course additional time is spent waiting for answers, although the time spent between Service Provider, Account Providers and Trust Manager equals 0.

The limiting factor in our system is the bandwidth of the connection of the mobile device. We found the built-in GPRS modem to be not suitable for our needs, therefore using the built in WiFi is the only solution for now. When comparing to the results of Svoboda et al. [30] (uncached) it is obvious that a HSDPA equipped device like the iPhone will be capable of delivering the necessary bandwidth.

Although for current mobile devices it is possible to handle the encryption necessary during the payment process as shown above, the encryption of the data places much higher loads on the device.

9. Conclusions and future work

As has been observed recently by Walburger et. al [31] there should be a chosen field of application and a respective business model the Mobile Grids. In our work we provide a concrete, secure business solution including accounting. There have been other attempts to connect mobile devices to the Grid (e.g. Amazon S3) but in our vision the Grid should embrace free markets where goods and services can be traded without restrictions. This includes that the Grid actually consists of a multitude of Service Providers, which compete for Clients. To enable this we propose a flexible standardized payment mechanism.

This work is an extension to our previous work where we introduced a ticket based infrastructure for the Mobile Grid. This ticketing mechanism makes our infrastructure different from other proxy based architectures, and is employed to control the number of accesses to distributed data and devices. In our chapter we have embedded the gSET accounting model in our algorithm and have shown that the inherent "trust aspect" of the gSET model leads to a simplified interaction between the business players. We have demonstrated that the gSET model is a very loosely coupled accounting model that can be embedded in a business workflow in a cohesive manner. Customer privacy in a commercial environment is an important issue. gSET maintains private information of Clients confidential. Nevertheless, a Service Provider has guarantees and can trust in client's characteristics.

Secure accounting and payment is a necessity to enable Grid Service in a mobile environment. Consuming Grid Services with a gSET account is comparable to payment by a personal credit card in many different shops but without disclosing the credit card number to them. In the future we plan to demonstrate the effectiveness of the gSET model as an enabling platform for commercial workflows in the Grid. For a standalone implementation of the gSET protocol see [32].

Acknowledgements

We want to express our thanks to the members of the BIG project team, who contributed to the ideas of this chapter: T. Weishäupl, F. Donno, H. Stockinger and E. Vinek. This work is partly supported by the Austrian Grid (Phase 2) project funded by the Austrian /bm:bwk (Federal Ministry for Education, Science and Culture), contract GZ BMWF-10.220/0002-II/10/2007, and by the OeNB Anniversary Fund, project no. 10547.

References

[1] Next Generation GRIDs Expert Group: Future for European Grids: GRIDs and Service Oriented Knowledge Utilities. Technical report, European Commission (January 2006).

[2] The gridbus project: Grid computing info centre (grid infoware) (2007).

[3] T. Weishaeupl, F. Donno, E. Schikuta, H. Stockinger, H. Wanek: Business In the Grid: The BIG Project. In: Grid Economics & Business Models (GECON 2005) of Global Grid Forum 13 (GGF13), Seoul, Korea (March 2005).

[4] T. Weishaeupl, C. Witzany, E. Schikuta: gSET: Trust Management and Secure Accounting for Business in the Grid. In: Sixth IEEE International Symposium on Cluster Computing and the Grid (CCGrid 2006), Singapore (May 16–19, 2006).

[5] ABI research: Mobile Cellular Industry as Set for the Year of the 3G Phone (2006).

[6] D. E. Millard, A. Woukeu, F. Tap, H. C. Davis: Experiences with Writing Grid Clients for Mobile Devices. In: 1st International ELeGI Conference, Napoli, Italy (March 14–16 2005).

[7] D. Chu, M. Humphrey: Mobile OGSI.NET: Grid Computing on Mobile Devices. In: 5th IEEE/ACM International Workshop on Grid Computing (Grid-04), Pittsburgh, PA, USA (Nov 8 2004).

[8] S. Kurkovsky, Bhagyavati, A. Ray, M. Yang: Modeling a Grid-Based Problem Solving Environment for Mobile Devices. In: International Conference on Information Technology: Coding and Computing (ITCC'04), Las Vegas, Nevada, USA (April 05–07 2004).

[9] Bhagyavati, S. Kurkovsky: Wireless Grid Enables Ubiquitous Computing. In: International Conference on Parallel and Disttributed Computing Systems (PDCS-2003), Reno, NV, USA (Aug 13–15 2003).

[10] C. Loos: E-Health with Mobile Grids: The Akogrimo Heart Monitoring and Emergency Scenario. White paper, University Hohenheim, Information Systems II, Stuttgart, Germany (Feb. 07 2006).

[11] T. Phan, L. Huang, C. Dulan: Challenge: Integrating Mobile Wireless Devices into the Computational Grid. In: 8th ACM International Conference on Mobile Computing and Networking (MobiCom). (Sept. 25–27 2002).

[12] U. Farooq, W. Khalil: A Generic Mobility Model for Resource Prediction in Mobile Grids. In: International Symposium on Collaborative Technologies and Systems (CTS-06), Las Vegas, Nevada, USA (May 14–17 2006).

[13] T. Weishaeupl, E. Schikuta: Towards the Merger of Grid and Economy. H. In Jin, Y. Pan, N. Xiao eds.: GCC Workshops. Volume 3252 of Lecture Notes in Computer Science., Springer (2004) 563–570.

[14] The Globus Security Team: Globus Toolkit Version 4 Grid Security Infrastructure: A Standards Perspective. (December 8, 2004).

[15] L. Pearlman, V. Welch, I. Foster, C. Kesselman, S. Tuecke: A Community Authorization Service for Group Collaboration. In: Proceedings of the IEEE 3rd International Workshop on Policies for Distributed Systems and Networks. (2002).

[16] Virtual Organization Membership Service (VOMS) (2003).

[17] H. Kishimoto: OGSA Status and Future. GGF13 OGSA-WG (March 14, 2005).

[18] A. Barmouta, R. Buyya: GridBank: A Grid Accounting Services Architecture (GASA) for Distributed Systems Sharing and Integration. In: 17th Annual International Parallel and Distributed Processing Symposium (IPDPS 2003) Workshop on

Internet Computing and E-Commerce, Nice, France, IEEE Computer Society Press (April 22–26, 2003) 245a.

[19] SweGrid: SweGrid Grid Accounting System (SGAS) (2005).

[20] I. Foster: What is the Grid? A Three Point Checklist. Technical report, Argonne National Laboratory and University of Chicago (July 2002).

[21] MasterCard, VISA: SET Secure Electronic Transaction Specification, Book 1: Business Description. (May 31, 1997) Version 1.0.

[22] MasterCard, VISA: SET Secure Electronic Transaction Specification, Book 2: Programmer's Guide. (May 31, 1997) Version 1.0.

[23] MasterCard, VISA: SET Secure Electronic Transaction Specification, Book 3: Formal Protocol Definition. (May 31, 1997) Version 1.0.

[24] H. R. Hansen, G. Neumann: Wirtschaftsinformatik 1, Grundlagen und Anwendungen. 9th edn. Lucius & Lucius, Stuttgart (2005).

[25] TeraGrid (2004).

[26] EU DataGrid Project (March 2004).

[27] W. Diffie, M. E. Hellman: New directions in cryptography. IEEE Transactions on Information Theory 22 (6) (1976) 644–654.

[28] O. Jorns, O. Jung, G. Quirchmayr: Transaction Pseudonyms in Mobile Environments. Springer Journal in Computer Virology (2007).

[29] Openmoko: Neo freerunner. Website (2008).

[30] P. Svoboda, F. Ricciato, W. Keim, M. Rupp: Measured web performance in gprs, edge, umts and hsdpa with and without caching. In: World of Wireless, Mobile and Multimedia Networks, 2007. WoWMoM 2007. IEEE International Symposium on a. (2007) 1–6.

[31] M. Waldburger, B. Stiller: Toward the Mobile Grid: Service Provisioning in a Mobile Dynamic Virtual Organizaion. In: IEEE International Conference on Computer Systems and Applications (AICCSA-06), Sharjah, Dubai (March 8–11 2006).

[32] T. Weishaeupl, C. Witzany: gSET (2005–2007).

Jürgen Mangler
Faculty of Computer Science
University of Vienna
Rathausstraße 19/9
1010 Vienna
Austria
e-mail: juergen.mangler@univie.ac.at

Erich Schikuta
Faculty of Computer Science
University of Vienna
Rathausstraße 19/9
1010 Vienna
Austria
e-mail: erich.schikuta@univie.ac.at

Christoph Witzany
Faculty of Computer Science
University of Vienna
Dr.-Karl-Lueger-Ring 1
1010 Vienna
Austria
e-mail: christoph.witzany@univie.ac.at

Oliver Jorns
Faculty of Computer Science
University of Vienna
Liebiggasse 4/3–4
1010 Vienna
Austria
e-mail: oliver.jorns@univie.ac.at

Irfan Ul Haq
Faculty of Computer Science
University of Vienna
Rathausstraße 19/9
1010 Vienna
Austria
e-mail: irfan.ul.haq@univie.ac.at

Helmut Wanek
Faculty of Computer Science
University of Vienna
Rathausstraße 19/9
1010 Vienna
Austria
e-mail: helmut.wanek@univie.ac.at

Part II:
Service Level Agreements

Economic Models and Algorithms for Distributed Systems, 107–108
Book Series: Autonomic Systems

Service Level Agreements

Macías et al. in their chapter "*Enforcing Service Level Agreements using an Economically Enhanced Resource Manager*" demonstrate how a resource management system can be extended with economic properties – primarily focusing on revenue gain and penalty determination when an SLA is violated. They demonstrate, using a scheduling example, the use of an EERM (Economically Enhanced Resource Manager) which utilizes these concepts to launch jobs on multiple resources. Various sub-systems within the EERM are described, along with discussion about what constitutes a violation, and the type of monitoring that is necessary to detect violations.

Püschel et al. in "*Extended Resource Management Using Client Classification and Economic Enhancements*" extend the work by Macías et al. by focusing on client classification that can be used to facilitate long standing interactions between specific users (e.g. standard vs. gold customers), thereby enabling better revenue management. They demonstrate how such economic considerations could be integrated with resource management using the EERM.

Smith and van Moorsel in their chapter "*Mitigating Provider Uncertainty in Service Provision Contracts*" indicate that it is often difficult for a provider to be precise about what capabilities it can deliver. Such uncertainly may arise due to inherent non-determinism within a distributed system, such as load fluctuations (in non-dedicated machines) and resource failures (which are, by definition, unpredictable). They discuss how such uncertainty subsequently manifests itself in a provider forming sub-optimal service contracts. They discuss the use of statistical estimators that could be used by a service provider to engage in the development of contracts. They also utilize customer classes in order to determine how service levels should be adapted when system properties change.

SLAs must utilize a term language in order to define: (i) what must be measured – and therefore whether an SLA was successfully completed or it failed; (ii) what the measured term means to an application end user. In this context, Tenschert and Kotsiopoulos in their chapter "*Text-Content-Analysis based on the Syntactic Correlations between Ontologies*" discuss how term languages can be mapped between different application domains using an "ontology" mapping process. They demonstrate that by utilizing greater semantic information using a Text-Content analysis phase, two interacting participants forming an SLA can un-

dertake better, and more targeted negotiation. They also demonstrate how such an approach could be used to complement an SLA lifecycle, and support better management of SLAs within multi-institutional collaborations.

Economic Models and Algorithms for Distributed Systems, 109–127
Book Series: Autonomic Systems

Enforcing Service Level Agreements Using an Economically Enhanced Resource Manager

Mario Macías, Garry Smith, Omer Rana, Jordi Guitart and Jordi Torres

Abstract. Traditional resource management has had as its main objective the optimisation of throughput, based on parameters such as CPU, memory, and network bandwidth. With the appearance of Grid Markets, new variables that determine economic expenditure, benefit and opportunity must be taken into account. The SORMA project aims to allow resource owners and consumers to exploit market mechanisms to sell and buy resources across the Grid. SORMA's motivation is to achieve efficient resource utilisation by maximising revenue for resource providers, and minimising the cost of resource consumption within a market environment. An overriding factor in Grid markets is the need to ensure that desired Quality of Service levels meet the expectations of market participants. This paper explains the proposed use of an Economically Enhanced Resource Manager (EERM) for resource provisioning based on economic models. In particular, this paper describes techniques used by the EERM to support revenue maximisation across multiple Service Level Agreements.

Mathematics Subject Classification (2000). Primary 91A40, 68M14, 68T99; Secondary 91A10.

Keywords. Grid computing, virtual organization, self organization, cooperative game theory

1. Introduction

The Self-organising ICT Resource Management (SORMA) [1] is an EU IST [2] funded project aimed at developing methods and tools for efficient market-based allocation of resources, using a self-organising resource management system and market-driven models, supported by extensions to existing grid infrastructure.

Topics addressed include Open Grid Markets, economically-driven middleware, and intelligent support tools.

Unlike traditional grid environments, jobs submitted to SORMA are matched with available resources according to the economic preferences of both resource providers and consumers, and the current market conditions. This means that the classic grid job scheduler, which is based on performance rules, is replaced by a set of self-organising, market-aware agents that negotiate Service Level Agreements (SLAs), to determine the 'best' resource allocation to fulfil both performance and business goals. In SORMA, an Economically Enhanced Resource Manager (EERM) exists at each resource provider's site, and acts as a centralised resource allocator to support business goals and resource requirements.

While a number of different economic models may be used to support resource management, this paper will focus on adaptation mechanisms to support revenue maximisation across multiple SLAs. In other words, when an EERM receives job/service reservations and associated SLAs, the EERM must allocate, monitor and enforce resource constraints in order maximise the number of jobs whose SLAs can be satisfied. However:

- the EERM does not have the ability to decide which jobs must be accepted or rejected. It is only used for consultative purposes. Even if the EERM advises that it cannot fulfill an incoming task, the economic agents could decide to send it to EERM to increase revenue.
- The EERM uses a predictive model to calculate the impact of a task execution. The prediction could be wrong, and the system would accept a job which could not be fulfilled, resulting in system overload.
- An abnormal situation could reduce the number of available resources; for example, some nodes of an available cluster could crash.

In the cases described before, the service provider would have a reduced number of resources, and the system becomes overloaded. The approach adopted in this work aims to minimise the economic impact of SLA violations, whilst at the same time attempting to enable as many jobs as possible to execute to completion.

The remainder of the paper is structured as follows: Section 2. presents related work. Section 3. defines a resource allocation scenario and describes revenue maximisation and SLA issues. Section 4. describes the EERM's architecture and highlights important features. Section 5. contains an example scenario that shows the EERM in action. Finally, Section 6. concludes the paper and describes our future work.

2. Related work

QoS has been explored in various contexts, such as for mobile devices [3] and multimedia applications [4]. Two types of QoS attributes can be distinguished: those based on quantitative, and qualitative resource characteristics. Qualitative

characteristics refer to aspects such as service reliability and user satisfaction. Quantitative characteristics refer to aspects such as network latency, CPU performance, or storage capacity. Although qualitative characteristics are important, it is difficult to measure these objectively. Systems which are centered on the use of such measures utilise user feedback [5] to compare and relate measures to particular system components. Our focus is primarily on quantitative characteristics.

Sahai et al. [6] propose an SLA management entity to support QoS in the context of commercial grids. They envision the SLA management entity existing within the Open Grid Services Architecture (OGSA), with its own set of protocols for manageability and assurance; they also describe a language for SLA specification. Although an interesting approach, this work is still at a preliminary stage, and the general applicability of this work is not obvious.

The Service Negotiation and Acquisition Protocol (SNAP) [7] is a resource management model to negotiate resources in distributed systems such as grids. SNAP defines three types of SLAs that coordinate management across a desired resource set, and can be used to describe complex distributed service requirements. Resource interactions are mapped to well-defined, platform-independent, SLAs with the SNAP protocol managing resources across different administrative domains, via three types of SLAs: Task SLA (TSLA), Resource SLA (RSLA) and Bind SLA (BSLA). The TSLA describes the task that needs to be executed, and the RSLA describes the resources needed to accomplish this task. The BSLA provides an association between the resources from the RSLA and the application 'task' in the TSLA. The SNAP protocol requires the existence of a resource management entity that can provide guarantees on resource capability; for example, RSLA.

Keahey et al. [8] propose an architecture called Virtual Application Service (VAS) for managing QoS in computational grids. VAS is an extended grid service with additional interfaces for negotiation of QoS level and service demands. The key objective of VAS is to facilitate the execution of real-time services with specific deadline constraints. A client submits a request to VAS for advance or immediate reservation of a service; supplying only time constraints. Essentially, VAS is a deadline-bound system, and the client can only specify time constraints as a QoS metric; VAS requires the application to predict how long it will need to run. Subsequently, VAS computes the time needed for service execution, based on a prediction model and service metadata.

In our approach we make use of different allocation strategies to run user applications, based on whether they require a *Time-domain* or *Resource-domain* allocation strategy. For users who request a Time-domain allocation, 100% of the computational resources must be allocated to their jobs. Whereas VAS requires users to benchmark their applications by running them first on an unloaded CPU, we utilise the results of application execution times where a guaranteed service execution has been requested, and use these as a benchmark. Burchard et al. [9] also propose the use of SLAs to negotiate service execution parameters between

resource managers. The SLA management is achieved via a Virtual Resource Manager (VRM). The VRM acts as a coordinator to aggregate SLAs negotiated with different sub-systems. Although the SLA management in this work is similar to our effort, the focus in our approach is on utilising the service paradigm, where the VRM is intended to integrate execution across a number of co-located clusters.

The General-purpose Architecture for Reservation and Allocation (GARA) [10] is the most commonly known framework to support QoS in the context of computational Grids. GARA allows users to specify end-to-end QoS requirements and provides advance reservations to various resources through a uniform interface. GARA's reservation is aimed at providing a guarantee that the client or application initiating the reservation will receive a specific QoS from the resource manager. Although GARA has gained popularity in the Grid community, it has limitations in coping with current application requirements and technologies, including: GARA is not OGSA-compliant; GARA does not support the concept of an agreement protocol to support the simultaneous allocation of resources; QoS monitoring and adaptation during the active QoS session is one of the most important and successful mechanisms to date in providing a quality guarantee [11], however, GARA does not provide adaptive functions to support this.

3. Scenario definition

Multiple economic enhancements exist that could be applied to resource management. In this paper we focus on only those related to revenue maximisation across multiple SLAs. However, an aim of our work is to provide a framework that will allow grid economists to define their own rules to achieve their particular goals. Therefore the content of this paper should be considered as a particular view of how the system behaves.

The **SLA Satisfaction Function** determines if, for a set of n resources $R = \{R_1, R_2, \ldots, R_n\}$, an SLA S can be fulfilled (results *true*) or will be violated (results *false*).

The **Multiple SLA Satisfaction Function** determines if, for a set of n resources $R = \{R_1, R_2, \ldots, R_n\}$, a set of m SLAs $\{S_1, S_2, \ldots, S_m\}$ can all be fulfilled or if any SLA will be violated.

Consider the follwing scenario – a set of running jobs each with its own SLA, is assigned to a resource. Each time a new job/SLA pair arrives, the EERM must assign a portion of the resource bundle. There are two possible scenarios:

- There are enough free resources, so the Multiple SLA Satisfaction Function is *true*. In this case, it is trivial to allocate the incoming tasks to a suitable resource. This scenario will not be studied in this paper.

- There are not enough resources (Multiple SLA Satisfaction function is *false*), implying that an intelligent resource re-allocation mechanism is required for maximising revenue and minimising SLA violation penalties.

3.1 Revenue maximisation in resource-limited providers

The **revenue** Rev_i is the amount of money that a client will pay if a provider fulfills the SLA S_i. The revenue is specified in the same SLA and usually has a fixed value. On the other hand, we define **penalty** Pen_i as the amount of money that the provider must pay if the SLA S_i is violated. The penalty is also specified in the same SLA and can be a function with parameters specified as in Section 3.2.

The **gain** $G(S_i)$ is the economic benefit that the provider obtains with the execution of a job whose SLA is S_i. It is defined as $G(S_i) = Rev_i - Pen_i$ and it can be positive (provider earns money) or negative (SLA violation with high penalty costs).

In a pool of resources R, executing a set of SLAs S at concrete instant t we define the **punctual gain** as:

$$\triangle G(t, R) = \sum_{i=1}^{m} G(S_i) = \sum_{i=1}^{m} Rev_i - \sum_{i=1}^{m} Pen_i$$

which is the gain (or loss) obtained if the current jobs all execute and finish on the resources that were assigned at instance t.

When a new SLA S_i arrives and there are not enough resources, system overload will cause the provider to start violating SLAs. To avoid (or minimise) violation penalties and maximise revenue, we suggest two complementary solutions:

- Dynamic adaptation in terms of resource provisioning. Previous work [12] has demonstrated that we can increase both the throughput and the number of jobs completed, by dynamically adapting the share of available resources between the applications by a function of demand. This is feasible when several applications share a single multi-processor platform (by assigning priorities and processors) or in virtualised environments [13], by dynamically assigning resources and priorities for each virtual machine).
- Task reallocation; finding a new resource assignation R' for each job i associated with the SLA S_i. The new gain will be defined as

$$\triangle G'(t, R) = \sum_{i=1}^{m} G'(S_i) - M(S, R)$$

where $M(S, R)$ is the economic cost of migrating the current running jobs S within the resource bundle R. When reallocating tasks, the main challenge for the EERM will be to find the highest $\triangle G'(t, R)$, by predicting the new gain for each possible assignment of resources, and trying to minimise the cost of resource reallocation $M(S, R)$.

3.2 SLA violation

Monitoring SLA Violation begins once an SLA has been defined. A copy of the SLA must be maintained by both the client and the provider. It is necessary to distinguish between an 'agreement date' (agreeing on an SLA) and an 'effective date' (subsequently providing a service based on the Service Level Objectives (SLOs) that have been agreed). A request to invoke a service based on the SLOs (which are the SLA terms), for instance, may be undertaken at a time much later than when the SLOs were agreed. During provision it is necessary to determine whether the terms agreed in the SLA have been met. In this context, a monitoring infrastructure is used to identify the difference between the agreed upon SLO and the value that was actually delivered during provisioning. It is also necessary to define what constitutes a violation. Depending on the importance of the violated SLO and/or the consequences of the violation, the provider in breach may avoid dispatch or obtain a diminished monetary sanction from the client.

An SLA may be terminated in three situations: (i) when the service defined in the SLA has completed; (ii) when the time period over which the SLA has been agreed upon has expired; and (iii) when the provider is no-longer available after an SLA has been agreed (for instance, the provider's business has gone into liquidation). In all three cases, it is necessary for the SLA to be removed from both the client and the provider. Where an SLA was actually used to provision a service, it is necessary to determine whether any violations had occurred during provisioning. As indicated above, penalty clauses are also part of the SLA, and need to be agreed between the client and the provider.

One of the main issues that the provider and the consumer will have to agree during the SLA negotiation is the penalty scheme or the sanctioning policies. Since both the service provider and the client are ultimately businesses (rather than consumers), they are free to decide what kind of sanctions they will associate to the various types of SLA breaches, in accordance with the weight of the parameter that was not fulfilled. We define the following broad categories of violation:

- 'All-or-nothing' provisioning: provisioning of a service meets all the SLOs – i.e. all of the SLO constraints must be satisfied for a successful delivery of a service;
- 'Partial' provisioning: provisioning of a service meets some of the SLOs – i.e. some of the SLO constraints must be satisfied for a successful delivery of a service;
- 'Weighted Partial' provisioning: provision of a service meets SLOs that have a weighting greater than a threshold (identified by the client).

Monitoring can be used to detect whether an SLA has been violated. Typically such violations result in a complete failure – making SLA violations an 'all-or-nothing' process. In such an event a completely new SLA needs to be negotiated, possibly with another service provider, which requires additional effort on both the client and the service provider. Based on this all-or-nothing approach, it is

necessary for the provider to satisfy all of the SLOs. This equates to a conjunction of SLO terms. An SLA may contain several SLOs, where some SLOs (e.g. at least two CPUs) may be more important than others (e.g. more than 100 MBytes of hard disk space). During the SLA negotiation phase, the importance of the different SLOs may be established. Clients (and service providers) can then react differently according to the importance of the violated SLO. In the WS-Agreement specification [14], the importance of particular terms is captured through the use of a 'Business Value'.

Weighted metrics can also be used to provide a flexible and fair sanction mechanism, in case an SLA violation occurs. Thus, instead of terminating the SLA altogether it might be possible to re-negotiate, i.e. with the same service provider, the part of the SLA that is violated. Again, the more important the violated SLO, the more difficult (if not impossible) it will be to re-negotiate (part of) the SLA.

4. Economically Enhanced Resource Manager

The overall aim of the EERM is to isolate SORMA economic layers from the technical ones and orchestrate both economic and technical goals to achieve maximum economic profit and resource utilisation. The main goals of the EERM are:

- To combine technical and economic aspects of resource management.
- Perform resource price calculations, taking into account current market supply and demand, performance estimations and business policies.
- To strengthen the economic feasibility of the Grid.

To provide a general solution that supports different scenarios and business policies, the EERM should provide flexibility in defining user (administrator) configurable rule-based policies, to support:

Individual Rationality An important requirement for a system is that it is individually rational on both sides, i.e. both providers and clients have to have a benefit from using the system. This is a requirement for the whole system, including features such as client classification or dynamic pricing.

Revenue Maximisation A key characteristic for SORMA providers is revenue (utility) maximisation. The introduced mechanisms can indeed improve the utility of both provider and client.

Incentive Compatibility Strategic behaviour of clients and providers can be prevented if a mechanism is incentive compatible. Incentive compatibility means that no other strategy results in a higher utility than reporting the true valuation.

Efficiency There are different types of efficiency. The first one considered here is that no participant can improve its utility without reducing the utility of another participant. The second efficiency criterion is allocative efficiency: i.e. the EERM must maximise the sum of individual utilities.

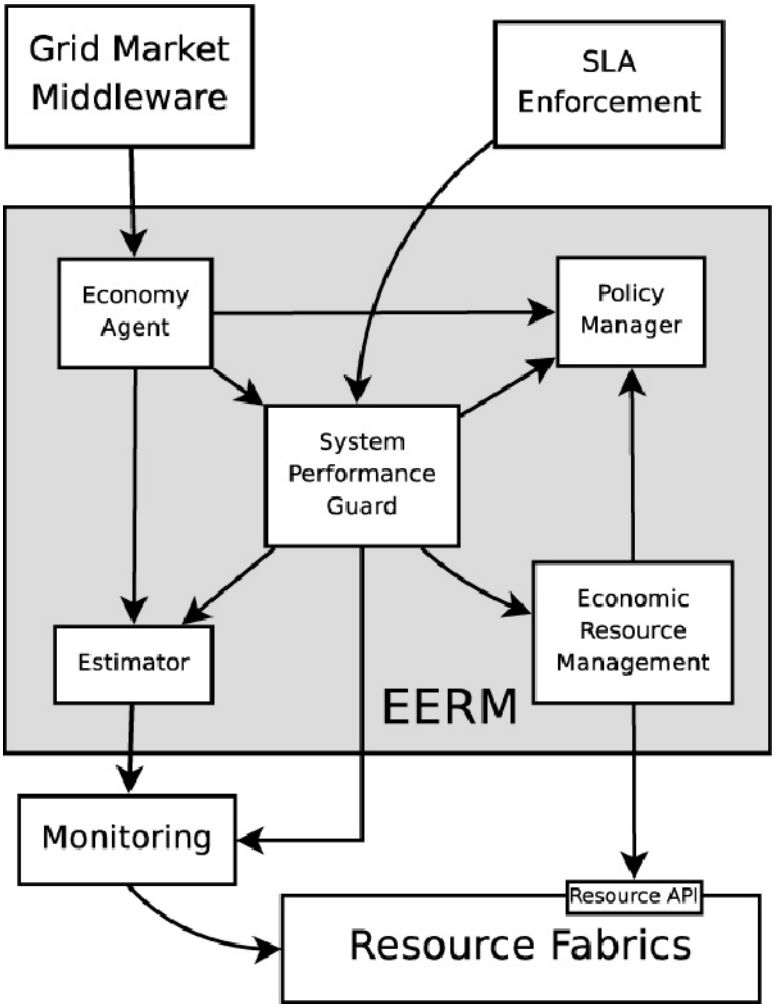

FIGURE 1. EERM components.

4.1 Architecture

The EERM's architecture is shown in Figure 1. To place the EERM in the context of the SORMA framework, we have also shown the SORMA Grid Market Middleware (GMM) [15], which provides the mechanisms to interact with the SORMA market. Once resource usage has been agreed in the SORMA market, a contract is sent to the EERM over the GMM. The contract provides the EERM with input for resource allocation, task execution and SLA enforcement activities. The EERM is comprised of the following components (see Figure 1):

Economy Agent (EA) The EA receives requests from SORMA market agents over the GMM. For each request, the EA checks whether the job is technically and economically feasible and calculates a price for the job based on the category of client (e.g. a preferred customer), resource status, economic policies, and predictions of future resource availability (provided by the Estimator Component). The EA interacts with the upper SORMA economic layers in the SLA negotiation process.

Estimator Component (EC) The EC calculates the expected impact on the utilisation of the Grid and is based on Kounev, Nou, and Torres [16]. In short, the EC's task is to avoid performance loss due to resource overload [12].

System Performance Guard (SPG) The SPG monitors resource performance and SLA violations. If there is a danger that one or more SLAs cannot be fulfilled, the SPG can take the decision of suspending, migrating or cancelling jobs to ensure the fulfilment of other, perhaps more important, SLAs with the aim of maximising overall revenue. Jobs can also be cancelled when additional capacity is required to fulfil commitments to preferred clients. The policies that dictate when to take action and which types of jobs should be killed, migrated or suspended are updated via the Policy Manager.

Policy Manager (PM) The PM stores and manages policies concerning client classification, job cancellation or suspension. Policies are formulated using the Semantic Web Rule Language (SWRL) [12]. The PM is an important part of the EERM in that it allows behaviour to be adapted at runtime. With the exception of the EC, all other EERM components use the PM to obtain policies that affect their decision making process.

Economic Resource Manager (ERM) The ERM interacts with local resource managers and is responsible for ensuring an efficient use of local resources. The ERM is described in further detail in Section 4.2.

Resource Monitoring (RM) The RM provides resource information for system and per-process monitoring. Resource information is used by the EC, SPG, ERM and SLA components. The RM is explained in further detail in Section 4.3.

SLA Enforcement (SLAE) The SLAE is tasked with monitoring SLA fulfillment. The SLAE uses monitoring data from the EERM and RM. When an SLA violation is detected, the SLAE takes reactive measures such as SLA renegotiation or compensation retrieval based on SLA penalty clauses. This component is explained in further detail in Section 4.4.

4.2 Economic Resource Manager (ERM)

The ERM (Figure 2) is designed to interact with a range of execution platforms (e.g. Condor, Sun Grid Engine, Globus GRAM, or UNIX fork) and achieves this using Tycho [18] connectors that communicate over the network to Resource Agents (RA).

The RA translates XML messages from the ERM into messages understood by the underlying platform (e.g. Condor). In addition, RAs provide a consistent

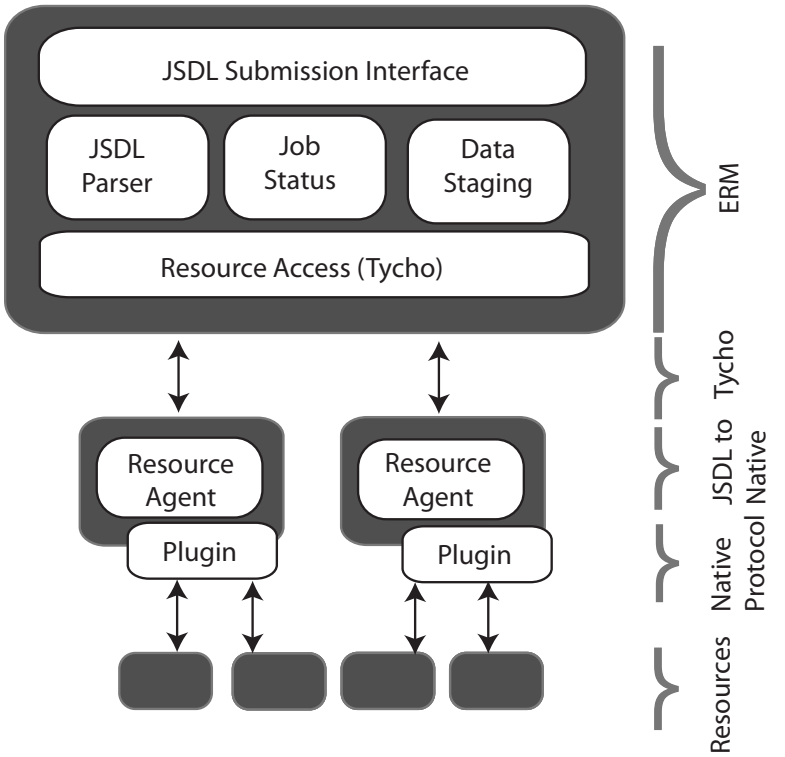

FIGURE 2. ERM implementation.

interface to the different underlying resource fabrics. This means that another platform can be adapted to SORMA by implementing an appropriate RA plug-in that performs translations to and from the underlying resource manager's native protocol. It is intended that access to the existing middleware be constrained by firewall rules, so that all interactions must go through the ERM. As a single point of access, the ERM can provide additional functionality that the underlying middleware may lack, for example, by providing support for advanced reservations.

In the current prototype, resource agents include a plugin for launching JSDL [19] jobs using GridSAM [20]. The approach used to implement the ERM is complimented by a similar approach used for resource monitoring.

4.3 Monitoring

In order to enable SLA enforcement, an understanding of the current and recent state of the underlying resources is required. Resource availability and utilisation can be sampled periodically in a coarse-grained manner in order to provide a high-level understanding of general Quality of Service (QoS) indicators. At other

times it may be appropriate to target particular and detailed attributes that reflect a given resources' ability to fulfil a particular action, e.g. the execution of a job. In addition notifications received from resources when a particular threshold has been exceeded can help to identify SLA violations. The EERM employs the GridRM [21] wide-area distributed monitoring system to gather data required for SLA enforcement.

The GridRM design employs gateways for gathering data from a number of different types of resources that make up the Grid. Resources of interest can include all manner of networked devices, from a remote sensor or satellite feed through to a computational node or a communications link. The Gateway is used internally, to a Grid-enabled site (the local layer), to configure, manage and monitor internal resources, while providing controlled external access to resource information. The EERM is bound to its local GridRM Gateway using the Tycho distributed registry and messaging system (see Figure 3). The EERM queries the gateway for real-time and historical resource data, and registers interest to receive different types of events that reflect changes in resource state (e.g. completion of a submitted job, system load greater than a specified threshold).

Resources may already provide legacy agents e.g. SNMP, Ganglia, /proc, Condor. As long as the Gateway is installed with a driver that supports the agent's native protocol, then all resource data provided by the native agent can be retrieved. In cases where an existing agent is not installed, a proprietary agent can be used for information gathering. Using a native agent means that existing resources can be monitored with little or no modification. Alternatively, installation of the proprietary GridRM agent implies some administrative overhead for each resource, but can result in improved performance and lower intrusiveness when gathering data.

Resource heterogeneity (agent and platform type) is hidden from GridRM clients and hence the EERM; the Structured Query Language (SQL) [22] is used to formulate monitoring requests, and a SORMA-specific schema based on the GLUE Schema [23] is used to group data and format the results into a consistent form (semantically and in terms of the values returned from different agents). Currently the SORMA consortium have identified a number of core attributes that are used for monitoring resources, enforcing SLAs, advertising resources on the market and for match making purposes. The core attributes include:

- CPU (architecture, number of, speed),
- Operating System (type, kernel version, shared libraries),
- Memory (total/free physical/virtual),
- Disk (total/free, network/local),
- Per-process execution statistics (start stop times, CPU time, memory footprint, exit status).

The current set of core attributes are a starting point and will evolve over time, as the requirements for more complex SLA enforcement are understood.

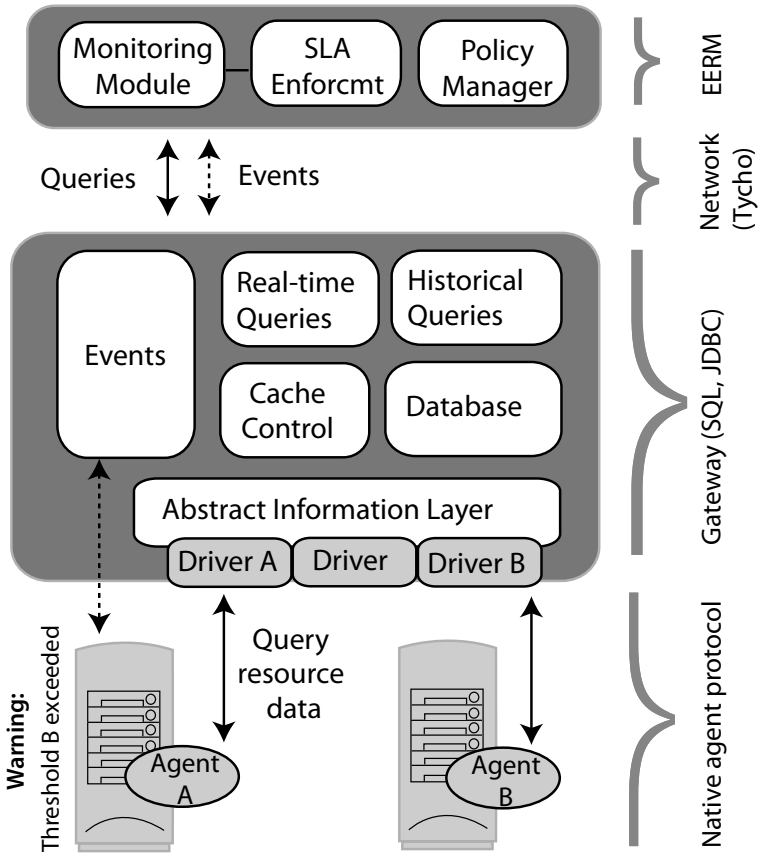

FIGURE 3. The GridRM gateway and relationship to the EERM.

As well as real-time information a need exists to capture historical data so that the SLA enforcement component can determine the likelihood of an SLA violation, based on past resource provision at a given site. The gateway can be instructed to query particular core attributes at a given frequency and store the results in its internal database. The consistent view of resource data provided by the GridRM gateway means that the SLA enforcement component is not exposed to resource heterogeneity and hence can focus on performing its core duties of SLA monitoring and enforcement.

4.4 SLA enforcement

The aim of the SLA enforcement component is to detect any SLA violation before it occurs, by evaluating the real time data about a provider to determine if a trend can be identified that fits in with a model that has shown, in the past using

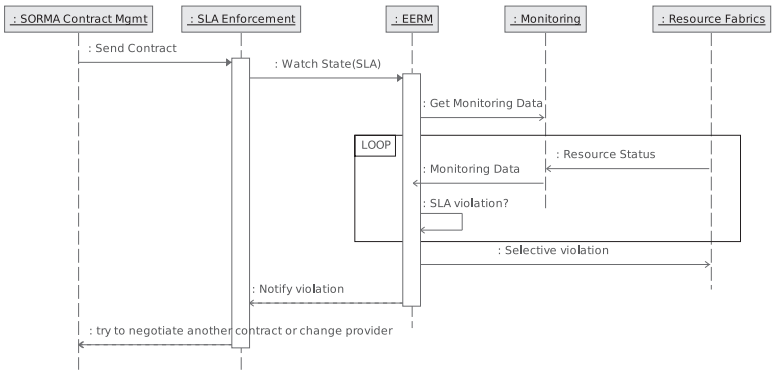

FIGURE 4. SLA enforcement and EERM components interaction.

historical data, to result in a possible violation. Since each SLA would consist of several resource attributes, the monitoring data collection will be for a metric that would represent collected statistical information about the resource providers past and current resource provision.

It is important to note that the SLA enforcement is not just for the providers to meet their commitments, it has to be monitored to validate that the consumers have met the SLA. One of the most important aspects to monitor that are relevant to consumers is the possibility of overuse of the resources than that agreed in the SLAs.

The interaction between the EERM and the SLA Enforcement component is described in Figure 4. The process begins when SLA Enforcement component receives a contract from SORMA Contract Management (the element which creates the contracts once a negotiation is agreed between providers and customers). After this, the SLA is created and sent to the EERM, which watches for its fulfillment. The EERM takes the economic data from SLA Enforcement and the performance data from Monitoring components to detect if an SLA is being violated, and performs a selective violation of SLAs to maximise the revenue.

On violation, the SLA Enforcement component detects this and generates a notification for the SORMA economic layers, in order to negotiate a new contract or give clients the possibility of searching for another provider.

5. Example scenario

To explain the operations of SLA fulfillment using the EERM, we have designed a simple conceptual scenario (see Figure 5): A resource provider wishes to sell the CPU time of four multi-processor machines. There are some free resources, and some running tasks whose revenues are specified in their SLAs. In order to simplify, there are two fixed economic parameters:

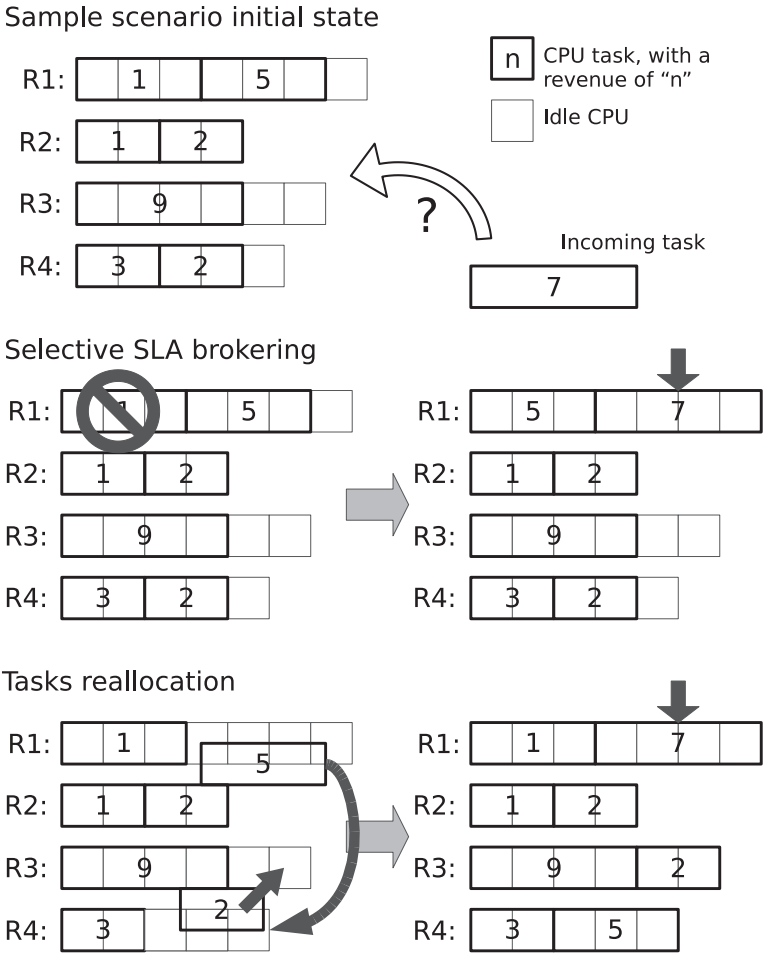

FIGURE 5. Enforcement of SLA fulfillment example scenario.

- Penalty for SLA violation: four currency units per violation, specified in each SLA.
- Task migration: one currency unit per migration; an indirect cost, calculated by the resource provider.

In the example scenario described in the upper schema of Figure 5, a new task arrives, and its SLA specifies a requirement for 4 CPUs and a revenue of 7. The incoming task does not fit in any resource, and therefore risks breaking the SLA. In response, the EERM could take three different actions:

1. Deny resource allocation for the incoming task. This is a non-economic response and means that the EERM has fallen back to the same behaviour as traditional resource management systems. Because the SLA has been agreed previously, if this response is taken the incoming task SLA will be broken and the provider will have to pay a penalty of 4. Using the formulas proposed in Section 3.1, the provider obtains a punctual gain of:

$$\triangle G(t,R) = \sum_{i=1}^{m} Rev_i - \sum_{i=1}^{m} Pen_i = 23 - 4 = \mathbf{19}\,.$$

2. Perform a selective SLA violation. In the middle schema of Figure 5, the EERM determines that the first task in R1 can be terminated due to the low revenue associated with that task. As a result the incoming task is now able to fit into R1. The punctual gain for the provider is:

$$\triangle G'(t,R) = 29 - 4 = \mathbf{25}\,.$$

3. Reallocate resources. In this particular case, there are 4 free CPUs, but they are scattered across the resource bundle. Reallocating tasks to provide a single machine with 4 CPUs may be cheaper than breaking the SLA. For example, the lower schema of 5, shows task migration which results in a new punctual gain of:

$$\triangle G'(t,R) = \sum_{i=1}^{m} G'(S_i) - M(S,R) = 30 - 2 = \mathbf{28}\,.$$

By applying economic enhancements into resource management, a provider can dramatically increase its revenue (47% in the example) by choosing the correct policy for SLA brokering or task reallocation. Determining the optimal solution for a given scenario will depend on penalty and reallocation costs as well as current resource availability.

6. Conclusions and future work

The work reported in this paper is motivated by the need to extend traditional resource management with economic parameters to support emerging grid markets. Within a grid market, resource providers must consider issues relating to current market conditions, QoS, revenue maximisation, economic sustainability and reputation, if they are to operate effectively.

In particular, this paper focuses on revenue maximisation using SLAs and describes how a strategic approach to managing SLAs can be used to secure optimal profit in situations where resources are scarce. Using our methods for selective SLA fulfilment and violation, the resource provider can determine which jobs should be

pre-empted in favour of freeing up resources for more lucrative SLAs. For example, it may be more profitable to violate an existing SLA, and pay the associated penalty, than it is to checkpoint and redistribute existing jobs, so that all SLAs can be fulfilled.

A prototype EERM is introduced and its architecture described. The EERM is a first attempt at providing strategic SLA enforcement within a grid market and forms part of the market mechanisms currently being implemented by the SORMA project.

Future work will include the identification of policies and parameters suitable for enforcing revenue maximisation given a number of different resource scenarios. The aim is to understand how to determine an optimal solution (or sub-optimal if the computation cost is too great) across a resource pool when complex policies and multiple economic parameters are at play. Another line of work will address how EERMs can be used to provide input to the market so that the negotiation process between customers and providers results in the generation of more accurate SLAs.

Acknowledgements

We would like to thank Mark Baker, SSE, University of Reading, for his comments during the writing of this paper. This work is supported by the Ministry of Science and Technology of Spain and the European Union (FEDER funds) under contract TIN2004-07739-C02-01 and Commission of the European Communities under IST contract 034286 (SORMA). Thanks are also due to Martijn Warnier and Thomas Quillinan of Vrije University, Amsterdam, The Netherlands – for discussions about types of violations that could arise in SLAs.

References

[1] Self-organizing ICT Resource Management (SORMA). [Online]. Available: http://www.sorma-project.eu.

[2] Information Society Technologies Programme. [Online]. Available: http://www.cordis.lu/ist.

[3] A. Oguz, A. T. Campbell, M. E. Kounavis, and R. F. Liao, "The Mobiware Toolkit: Programmable Support for Adaptive Mobile Networking", *IEEE Pesronal Communications Magazine, Special Issue on Adapting to Network and Client Variability*, 1998.

[4] G. Bochmann and A. Hafid, "Some Principles for Quality of Service Management", Tech. Rep., 1996.

[5] V. Deora, J. Shao, W. A. Gray and N. J. Fiddian, "A quality of service management framework based on user expectations", *First International Conference on Service Oriented Computing* (*ICSOC*), *Trento, Italy*, December 2003.

[6] A. Sahai, S. Graupner, V. Machiraju and A. Moorsel, "Specifying and Monitoring Guarantees in Commercial Grids through SLA", *Proceedings of the 3rd IEEE/ACM CCGrid* 2003, 2003.

[7] K. Czajkowski, A. Dan, J. Rofrano, S. Tuecke and M. Xu, "Agreement-based Grid Service Management (OGSI-Agreement)", *Grid Forum, GRAAP-WG Author Contribution Draft*, June 2003.

[8] K. Keahey and K. Motawi, "The Taming of the Grid: Virtual Application Service", *Argonne National Laboratory Technical Memorandum No.* 262, May 2003.

[9] L. Burchard, M. Hovestadt, O. Kao, A. Keller and B. Linnert, "The virtual resource manager: An architecture for sla-aware resource management", *In Proceedings of IEEE CCGrid* 2004*, Chicago, US*, 2004.

[10] I. Foster, C. Kesselman, C. Lee, R. Lindell, K. Nahrstedt and A. Roy, "A distributed resource management architecture that supports advance reservation and co-allocation", *In Proceedings of the International Workshop on Quality of Service*, pp. 27–36, 1999.

[11] R. Al-Ali, A. Hafid, O. Rana and D. Walker, "An Approach for QoS Adaptation in Service-Oriented Grids", *Concurrency and Computation: Practice and Experience Journal*, vol. 16, no. 5, pp. 401–412, 2004.

[12] R. Nou, F. Julià, J. Guitart and J. Torres, "Dynamic resource provisioning for self-adaptive heterogeneous workloads in smp hosting platforms", *ICE-B* 2007*, International Conference on E-Business, Barcelona, Spain*, July 2007.

[13] D. Gupta, L. Cherkasova, R. Gardner and A. Vahdat, "Enforcing performance isolation accross virtual machines in xen", *Middleware* 2006*, Melbourne, Australia*, Nov. 27–Dec. 1 2006.

[14] Web Services Agreement specification. [Online]. Available: http://www.ogf.org/documents/GFD.107.pdf.

[15] L. Joita, O. F. Rana, P. Chacín, I. Chao, F. Freitag, L. Navarro and O. Ardaiz, "Application deployment using catallactic grid middleware", *Middleware for grid computing*, 2005.

[16] S. Kounev, R. Nou and J. Torres, "Using QPN to add QoS to Grid Middleware", Universitat Politècnica de Catalunya, Tech. Rep., 2007.

[17] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof and M. Dean, "Swrl: A semantic web rule language combining owl and ruleml", W3C Member submission 21 may 2004, Tech. Rep., 2004. [Online]. Available: http://www.w3.org/Submission/SWRL/.

[18] M. Baker and M. Grove, "A virtual registry for wide-area messaging", *IEEE International Conference on Cluster Computing*, September 2006.

[19] Job Submission Description Language (JSDL) Work Group. [Online]. Available: http://forge.gridforum.org/projects/jsdl-wg.

[20] GridSAM, Grid Job Submission and Monitoring Web Service. [Online]. Available: http://gridsam.sourceforge.net/.

[21] M. Baker and G. Smith, "Gridrm: an extensible resource monitoring system", *IEEE International Conference on Cluster Computing*, 2003.

[22] A. Eisenberg, J. Melton, K. Kulkarni, J. Michels and F. Zemke, "Sql: 2003 has been published", *SIGMOD Record*, vol. 33, no. 1, March 2004.

[23] S. Andreozzi, "Glue schema implementation for the ldap data model", Instituto Nazionale Di Fisica Nucleare, Tech. Rep., September 2004.

Mario Macías
Technical University of Catalonia
Barcelona Supercomputing Center
c/ Jordi Girona 29
08034 Barcelona
Spain
e-mail: mario.macias@bsc.es

Garry Smith
School of Systems Engineering
University of Reading
RG6 6BX
United Kingdom
e-mail: garry.smith@computer.org

Omer Rana
School of Computer Science
Cardiff University
CF24 3AA
United Kingdom
e-mail: O.F.Rana@cs.cardiff.ac.uk

Jordi Guitart
Technical University of Catalonia
Barcelona Supercomputing Center
c/ Jordi Girona 29
08034 Barcelona
Spain
e-mail: jguitart@ac.upc.edu

Jordi Torres
Technical University of Catalonia
Barcelona Supercomputing Center
c/ Jordi Girona 29
08034 Barcelona
Spain
e-mail: jordi.torres@bsc.es

Economic Models and Algorithms for Distributed Systems, 129–141
Book Series: Autonomic Systems

Extended Resource Management Using Client Classification and Economic Enhancements

Tim Püschel, Nikolay Borissov, Dirk Neumann, Mario Macías, Jordi Guitart and Jordi Torres

Abstract. Commercialization of computing resources will become more and more important as the transition from Grid computing in academic environments to commercial services based on concepts such as utility or Cloud computing progresses. This results in the necessity to not only base components on technical aspects, but also to include economical aspects in their design. This paper presents a framework that links technical and economical aspects to the management of computational resources. Economic enhancements like dynamic pricing and client classification are introduced based on a technical resource management environment and positioned within this resulting in a proposed architecture for an Economically Enhanced Resource Manager (EERM). The introduced approach is evaluated considering various economic design criteria and example scenarios.

Mathematics Subject Classification (2000). Primary 91A40, 68M14, 68T99; Secondary 91A10.

Keywords. Grid computing, virtual organization, self organization, cooperative game theory

1. Introduction

In many cases IT applications – such as data mining, portfolio analysis and video stream analysis – are characterized by the fact that they have a strongly varying demand for resources like processors and storage. To accommodate peak load times, it is necessary to maintain an adequate IT-infrastructure. During off-peak times these resources mainly remain idle. With increasing global competition, enterprises are forced to cut down costs dramatically and therefore strive for trimming down costs for IT infrastructure [5].

This need gave rise to the vision of utility computing [21] where computer resources can be accessed dynamically in analogy to electricity and water. Utility computing becomes more powerful if more resource providers add their resources to the Grid. It is thus the utmost objective to attract more providers. The concept of Grid Computing has lost some attention to newer concepts such as Cloud Computing in recent years. However the issues adressed in this work still remain relevant in light of more recent concepts such as Cloud Computing.

With state-of-the-art technology, this assimilation is hampered, as the local resource managers facilitating the deployment of resources are not designed to incorporate economic issues (e.g. price). They plainly adhere to technical parameters that define when jobs are scheduled. In the easiest case, the local resource managers employ a First-in-First-out algorithm for scheduling ignoring all economic factors.

In recent times, several research projects have started to develop price-based resource management components that support the idea of utility computing. Those approaches are entirely devoted to scheduling by utilizing the price mechanism.

In addition, resource management is much more comprehensive than just scheduling. For example Service-Level-Agreement (SLA) management is also part of resource management that is oftentimes omitted in economic approaches. This plays for instance a role when deciding which already ongoing jobs to cancel in overload situations in order to maintain system stability. To achieve better performance in the commercialization of distributed computational resources, decisions about the supplied resources and their management therefore should be based on both on a technical and on an economic perspective [13].

Technical resource management systems typically offer the possibility to include priorities for user groups. In purely price-based schedulers it is not possible to distinguish important from unimportant partners, as only current prices matters for the allocation.

2. Objectives

The objectives of this work are to motivate and introduce economical enhancements to resource management and present an architecture comprising these enhancements. We will motivate that client classification should be integrated into economically enhanced resource management systems.

Essentially, there are two main reasons to do so: First, client classification allows the inclusion of long-term oriented relationships with strategically important customers. Second, client classification can be used as an instrument of revenue management, which allows skimming off consumer surplus. The main contribution of this paper is to show how technical parameters can be combined into an economically enhanced resource management that increases revenue for the local resource sites.

The remainder of the paper is structured as follows: Section 3 presents a motivational scenario, Section 4 related work. Section 5 explains the economic enhancements and the mechanism of client classification. Section 6 gives an overview of the goals and the architecture of the EERM. Subsequently Section 7 contains an example scenario and a short evaluation of the proposed mechanisms. Finally, Section 8 concludes the paper and describes our future work.

3. Motivational scenario

Suppose a large service provider maintains an IT-Infrastructure, whose free resources are offered to external users. The service provider already has a number of clients but depending on the time specific resource usage, it has fluctuating spare capacity. Therefore, the service provider offers its excessive resources over a Grid market to find new clients and optimize the resource utilization and thus its revenue. To retain the good relations with current clients and encourage regular use of its services, the provider offers special-agreements to preferred client. Preferred clients (Gold-clients) receive preferences when submitting jobs and obtain reservation on certain share of the provider's resources.

For clients who do not want to entrust their data to just any arbitrary Grid service provider, this scenario is a very interesting option. They can have preferred client contracts with a few selected Grid service providers. This increases their chances to access the services of a trusted provider on demand. Preferred clients are also awarded with further benefits, such as better service levels or price discounts, from the providers as additional incentives to use their resources.

At the same time the proposed mechanisms are not exclusive, so they leave the option to use other Grid resources of other providers in case of missing free capacity.

4. Related work

Requirements on quality of service (QoS) functionalities by the management of disperse computational resource are considered into the Globus Architecture for Reservation and Allocation (GARA) [11]. The incorporated components are a first step to achieving end-to-end QoS guarantees by introducing advanced reservation of resources.

QoS can also be achieved by introducing risk management into the Grid, which is elaborated in [7]. Their proposal allows modeling and managing the risk that the service level agreement (SLA) cannot be fulfilled. This allows taking the risk of SLA failure into account when deciding on prices and penalties.

Another approach for autonomic quality of service aware resource management is based on online performance models [14]. The authors introduce a framework for designing resource managers that are able to predict the impact of a job

in the Grid performance and adapt the resource allocation in such a way that service level agreements (SLAs) can be fulfilled.

Elements of client classification in Grids such as price discrimination, based on customer characteristics, are explored in [3] and [17]. Chicco et al. describe data-mining algorithms and tools for client classification in the electricity Grids [6] but concentrate on methods for finding groups of customers with similar behaviors.

Further related work in client classification includes a framework for admission control on e-commerce websites that prioritizes user sessions based on predictions about the user's intentions to buy a product [19].

5. Economic enhancements and client classification

One key requirement of commercial resource managers is to offer QoS regarding the job execution, such as guarantees about the available resources and execution time. The first objective of an economic enhanced resource manager is to maximize its revenue, e.g. by allocating as many jobs as possible. However overload situations can lead to reduced overall performance [18] and break QoS agreements between the provider and clients.

To avoid this, the resource manager needs information about the current utilization of the offered resources as well as information about the required resource capacity of the incoming jobs. While the information about the current utilization could be delivered from monitoring services, the jobs execution time and thus its required capacity according to the jobs execution-deadline is difficult to estimate. Kounev et al. [14] propose a mechanism to estimate the influence of a jobs execution on the utilization through online performance models. The agreed QoS also should be met when some of the computational resources fail. This requires to keep an adequate percentage of the offered resources free, so failure situations can be handled transparent to the client and their jobs QoS still met. Where such a buffer is not possible or desired it is crucial to at least meet as many SLAs as possible and thereby minimize the negative effect of failure situations.

To this end we propose the introduction of a Job Cancellation and Suspension mechanism. In case predictions are inaccurate or problems in the Grid lead to reduced capacity some jobs are cancelled or suspended to ensure the other jobs meet their SLA. The decision which jobs to cancel should be taken in such a way that the overall revenue is maximized, i.e. the cancellation penalties and the loss of revenue due to cancelled jobs are minimized.

As another enhancement we introduce dynamic pricing based on technical as well as economic factors. Resource prices can be based on the current utilization of the Grid, the impact a job has on the utilization of the Grid, client classification as well as other factors such as current supply and demand. For example when the resource utilization is very high or an incoming job leads to a high utilization a higher price is charged.

A client is a user or an application which is willing to allocate and consume distributed computational resources. The following factors can be used to differentiate client classes:

- Price discrimination: is an economic factor proposed in the past by different authors. One example is the idea of using Grid miles [3] [17], in analogy to airlines' frequent flyer miles. In general certain clients can be given price discounts on reservation or final prices.
- Job prioritization is another option to differentiate clients. We differentiate two types of job-priorities strict and soft priorities.
  - Strict priority means that jobs from clients with priority are always preferred over clients without priority. Thus there is no real competition between the different classes of clients.
  - Soft priority means jobs from clients with priority are generally preferred but clients without priority have the chance to outbid prioritized clients.
- Reservation of resources is important for clients who want to ensure that they always have a certain capacity at their disposal.
- The last introduced discrimination factor is quality of service (QoS) which results in versioning of the service as known from pricing theory [23]. Offering different levels of QoS for different classes of clients is possible by modeling them in the SLA.

Based on the described desired properties of the resource manager as well as the abovementioned client classification factors we introduce a framework of economic enhanced resource management.

6. Economically Enhanced Resource Management

Beside the specified economic enhancements regarding the client classification the EERM-mechanism has to satisfy common economic design criteria explained in the first subsection. To allow the integration of client classification, associated economic enhancements, as well as the economic design criteria we propose and describe a framework of Economically Enhanced Resource Manager (EERM).

6.1 Economic design criteria

The EERM has to satisfy following economic design criteria proposed in [4] and [22]:

- *Individual rationality*: The provider has to have a benefit from using the EERM as well the client should benefit from choosing a provider using the EERM. For the provider this benefit could be a higher or more predictable revenue, lower risk (e.g. of paying penalties) and better client retention. For the client this benefit could be a higher ratio of acceptance of important jobs, lower prices, better service levels or preferred acceptance of jobs.

- For the criterion of *incentive compatibility* it is important to choose the characteristics of the mechanism in such a way that the clients report their true requirements. This avoids strategic behavior, e.g. with the aim to influence the client classification.
- *Revenue maximization*: The objective of the resource providers is to maximize their revenue, which is one of the economic characteristics of the EERM.
- Client Classification adds some additional *computational complexity*. Depending on the policies that are chosen winner determination has to be slightly adapted. It, however, does not introduce any NP-hard problems into the mechanism and the additional computational cost should be limited.
- Another criterion is *efficiency*. A mechanism is called allocative efficient if it maximizes the sum of individual utilities.

6.2 Model of the EERM

The main goals of the EERM are to:

- link technical and economical aspects of resource management
- establish more precise price calculations for resources, taking usage of the Grid, performance estimations and business policies into account and
- strengthen the economic feasibility of the Grid

In this work we focus on presenting the features of the EERM that can be related to client classification.

Figure 1 shows the Architecture of the EERM. The Economy Agent first receives a request from a market agent, checks whether the job is technically and economically feasible and calculates a price for the job based on the clients class, resource availability, pricing policies as well as predictions of future job executions from the estimator component.

The Estimator component calculates the expected impact on the utilization of the Grid.

The System Performance Guard (SPG) monitors the performance of the providers supplied resources and ensures the accepted SLAs. If there is a risk that one or more SLAs cannot be fulfilled the SPG can take decisions to suspend or cancel jobs to ensure the fulfillment of the SLAs and maximize overall revenue.

To keep the EERM adaptable, the Policy manager stores and manages policies for client classification, job cancellation and suspension. Policies are formulated in the Semantic Web Rule Language (SWRL) [12]. An example for a policy based rule is like the following:

$$ClientClass(?clientclass) \wedge sameAs(?clientclass, \text{“}Standard\text{”}) \wedge$$

$$Utilization(?utilization) \wedge InsideUtilizationRange(?utilization, \text{“}70\% - 100\%\text{”})$$

$$\Rightarrow RejectJob\,.$$

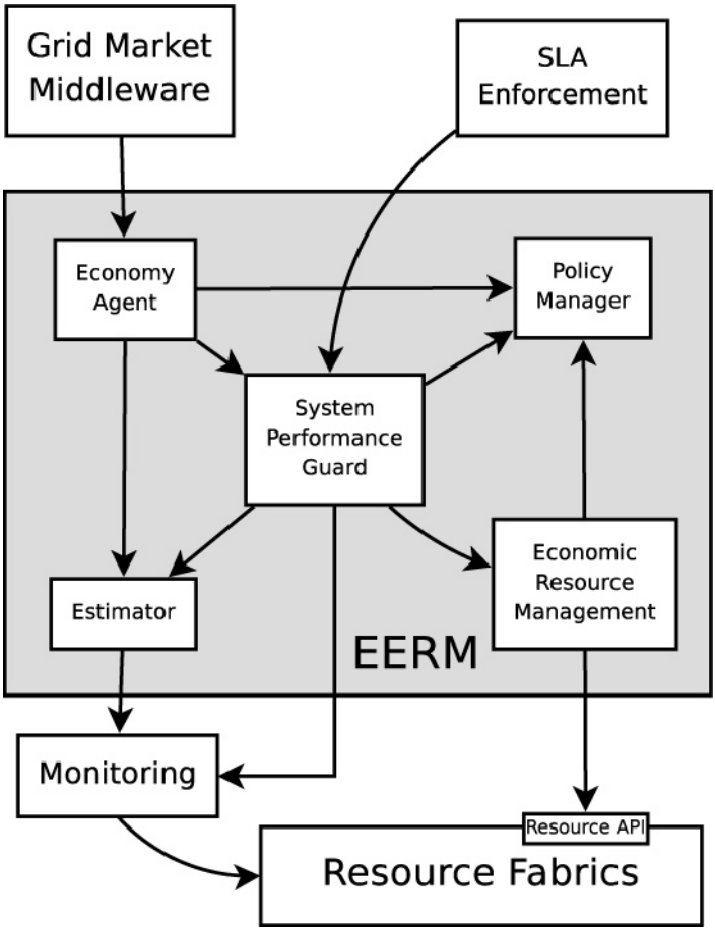

FIGURE 1. EERM architecture.

This policy express that if the utilization of the Grid is between 70% and 100% and the client classification of a job is Standard the job is not accepted. This implies that in this case only jobs with other classifications are accepted.

The Economic Resource Manager is responsible for the communication with the local resource managers and influences the local resource management to achieve a more efficient global resource usage.

The EERM interacts with various other components, namely a Grid Market Middleware, a Monitoring component and the Resource Fabrics. The Grid Market Middleware represents the middleware responsible for querying prices and offering of resources.

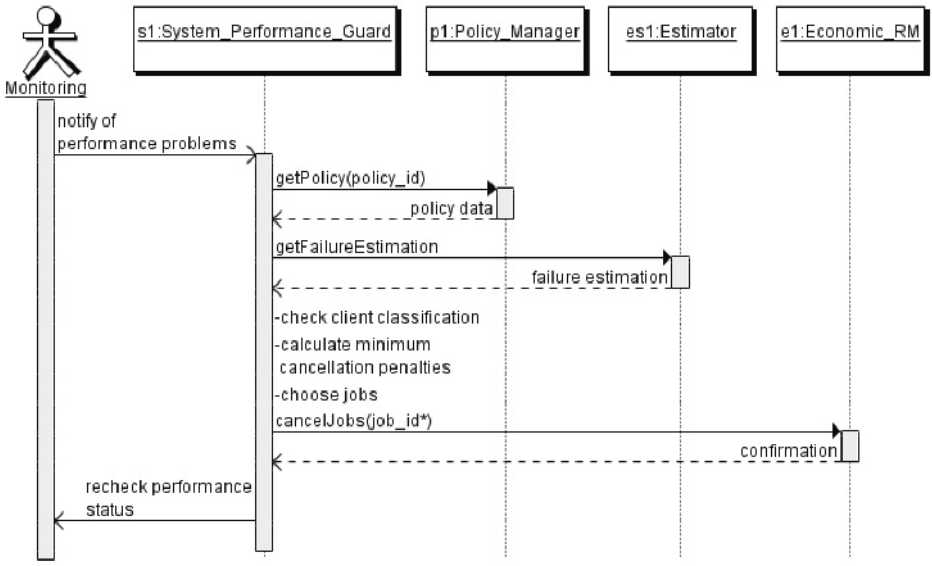

FIGURE 2. Sequence diagram of job cancellation.

The Monitoring is responsible for monitoring the state and the performance of the Grid.

Resource Fabrics enables the low level access to the Grid-resources, e.g. via Condor [16] or Globus [10].

Figure 2 shows the sequence of a Job Cancellation due to performance problems on the Grid. First the Monitoring informs the System Performance Guard about problems on the grid. Then the SPG proofs the policies, request the necessary information, chooses the jobs to be cancelled and interacts finally with the Economic RM to initiate the cancellation of the respective jobs.

7. Evaluation

For the example scenario we have several assumptions:

- We assume that the system only receives information about jobs which become available in the following time period.
- A Gold-client only uses the provider if he can launch jobs with capacity requirements between 30 units and 60 units per period.
- The total capacity of the provider is 100.

The jobs shown in Table 1 will be available during the run.

First we consider a scenario without EERM (Case I). In this case any job is accepted if there is enough capacity left to fulfill it. As can be seen in Table 2 Jobs A, B, C, G, H, L, M, O, P, and Q are accepted, the other jobs cannot be

Table 1. Example data with available jobs during the run.

Jobs	Start	End	Capacity/t	Client Class	Price
A	1	3	55	Standard	330
B	1	5	24	Standard	180
C	1	7	20	Standard	140
D	2	4	20	Standard	120
G	4	7	20	Standard	160
H	4	9	15	Standard	135
L	6	8	30	Standard	90
M	6	9	12,5	Standard	50
O	8	10	20	Standard	90
P	9	10	21	Standard	84
Q	9	10	30	Standard	90
R	2	6	30	Gold	375
S	5	8	30	Gold	300
T	7	10	7,5	Gold	75
U	7	9	20	Gold	150
V	9	10	20	Gold	100

Table 2. Allocation example data with available jobs during the run.

Case	Jobs completed	Revenue	Avg Utilization
I	A, B, C, G, H, L, M, O, P, Q	1349	89.7
II	C, D, R, S, M, T, U, O, V	1400	71.2
III	A, R, G, H, S, T, U, V	1625	73.5

accepted due to capacity constraints. This results in total revenue of 1349 and an average utilization of 89.7.

Then we introduce a fixed reservation of 60% of the resources for the Gold-client to ensure that his requirements are fulfilled (Case II). Now the total revenue is 1400 and an average utilization of 71.2. Even though the utilization is lower an increase in revenue can be achieved.

In the third case the policy is to accept only jobs from the Gold-client if the job would result in utilization higher than 70%. In case this is not sufficient to fulfill the requirements of the Gold-client, jobs from Standard-clients are stopped. This is a policy that can be used in situations as described in the motivational scenario. In this case it is not necessary to stop any jobs and the policy results in total revenue of 1625 and an average utilization of 73.5. This is a significant increase in revenue.

8. Conclusions

In this paper we motivated client classification and further economical enhancement for resource management. We presented factors and technical parameters that can be used for these enhancements to increase revenue for the local resource sites. Furthermore we introduced the preliminary architecture for an Economically Enhanced Resource Manager integrating these enhancements. Due to the general architecture and the use of policies and a policy manager our approach is can be adapted to a wide range of situations.

We evaluated our approach considering economic design criteria and using an example scenario. The evaluation of our first model shows that the proposed economic enhancements firstly enable maximizing providers benefit and secondly strengthen the relationship with business clients.

The next steps will include refinement of the architecture as well as the implementation of the EERM. During and following this process, further evaluation of the system will be done, e.g. by testing the system and running simulations. Another issue that requires further consideration is the autonomous generation of business policies for the EERM.

Acknowledgements

This work is supported by the Ministry of Science and Technology of Spain and the European Union (FEDER funds) under contract TIN2004-07739-C02-01 and Commission of the European Communities under IST contract 034286 (SORMA).

References

[1] S. Aiber, D. Gilat, A. Landau, N. Razinkov, A. Sela and S. Wasserkrug (2004). Autonomic self-optimization according to business objectives. In *ICAC'04: Proceedings of the First International Conference on Autonomic Computing* (*ICAC'04*), Washington, DC, USA, pp. 206–213. IEEE Computer Society.

[2] H. Boughton, P. Martin, W. Powley and R. Horman (2006). Workload class importance policy in autonomic database management systems. In *POLICY'06: Proceedings of the Seventh IEEE International Workshop on Policies for Distributed Systems and Networks* (*POLICY'06*), Washington, DC, USA, pp. 13–22. IEEE Computer Society.

[3] R. Buyya (2002). *Economic-based Distributed Resource Management and Scheduling for Grid Computing*. Ph. D. thesis, Monash University.

[4] D. E. Campbell (1987). *Resource Allocation Mechanisms*. Cambridge University Press, London.

[5] N. G. Carr (2005). The end of corporate computing. *MIT Sloan Management Review* 46(3), 32–42.

[6] G. Chicco, R. Napoli and F. Piglione (2006). Comparisons among clustering techniques for electricity customer classification. *IEEE Transactions on Power Systems* 21(2), 933–940.

[7] K. Djemame, I. Gourlay, J. Padgett, G. Birkenheuer, M. Hovestadt, O. Kao and K. Voß (2006). Introducing risk management into the grid. In *The* 2*nd IEEE International Conference on e-Science and Grid Computing* (*eScience*2006), Amsterdam, Netherlands, pp. 28.

[8] D. G. Feitelson (2007). *Workload Modeling for Computer Systems Evaluation.*

[9] D. F. Ferguson, C. Nikolaou, J. Sairamesh and Y. Yemini (1996). Economic models for allocating resources in computer systems. pp. 156–183.

[10] I. Foster and C. Kesselman (1997). Globus: A metacomputing infrastructure toolkit. *International Journal of Supercomputer Applications and High Performance Computing* 11(2), 115–128.

[11] I. Foster, C. Kesselman, C. Lee, B. Lindell, K. Nahrstedt and A. Roy (1999). A distributed resource management architecture that supports advance reservations and co-allocation. In *Proceedings of the* 7*th International Workshop on Quality of Service* (*IWQoS* 1999), London, UK, pp. 62–80.

[12] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof and M. Dean (2004). Swrl: A semantic web rule language combining owl and ruleml. w3c member submission.

[13] C. Kenyon and G. Cheliotis (2004). Grid resource commercialization: economic engineering and delivery scenarios. *Grid resource management: state of the art and future trends*, 465–478.

[14] S. Kounev, R. Nou and J. Torres (2007a). Autonomic qos-aware resource management in grid computing using online performance models. In *The* 2*nd International Conference on Performance Evaluation Methodologies and Tools* (*VALUETOOLS* 2007), Nantes, France.

[15] S. Kounev, R. Nou and J. Torres (2007b). Building online performance models of grid middleware with fine-grained load-balancing: A globus toolkit case study. In *The* 4*th European Engineering Performance Workshop* (*EPEW* 2007), Berlin, Germany.

[16] M. J. Litzkow, M. Livny and M. W. Mutka (1988). Condor – A hunter of idle workstations. In *Proceedings of the* 8*th International Conference of Distributed Computing Systems* (*ICDCS* 1988).

[17] S. Newhouse, J. MacLaren and K. Keahey (2004). Trading grid services within the uk e-science grid. *Grid resource management: state of the art and future trends*, 479–490.

[18] R. Nou, F. Julià and J. Torres (2007). Should the grid middleware look to self-managing capabilities? In *The* 8*th International Symposium on Autonomous Decentralized Systems* (*ISADS* 2007), Sedona, Arizona, USA, pp. 113–122.

[19] N. Poggi, T. Moreno, J. L. Berral, R. Gavaldà and J. Torres (2007). Web customer modeling for automated session prioritization on high traffic sites. In *Proceedings of the 11th International Conference on User Modeling*, Corfu, Greece.

[20] T. Püschel, N. Borissov, M. Macías, D. Neumann, J. Guitart and J. Torres (2007). Economically enhanced resource management for internet service utilities. In *Proceedings of the 8th International Conference on Web Informations Systems Engineering* (*WISE* 2007), pp. 335–348.

[21] M. A. Rappa (2004). The utility business model and the future of computing services. *IBM Systems Journal* 43(1), 32–42.

[22] P. R. Wurman (1999). *Market structure and multidimensional auction design for computational economies.* Ph. D. thesis, University of Michigan. Chair-Michael P. Wellman.

[23] H. R. Varian (1997). *Versioning Information Goods.* University of California, Berkeley, USA

[24] C. S. Yeo and R. Buyya (2004). Pricing for Utility-driven Resource Management and Allocation in Clusters. In *Proceedings of the 12th International Conference on Advanced Computing and Communications* (*ADCOM* 2004), Ahmedabad, India, pp. 32–41.

Tim Püschel
Chair for Information Systems Research
Albert-Ludwigs-Universität Freiburg
Kollegiengebäude II
Platz der Alten Synagoge
79085 Freiburg
Germany
e-mail: tim.pueschel@is.uni-freiburg.de

Nikolay Borissov
Institute of Information Systems and Management (IISM)
Universität Karlsruhe (TH)
Englerstraße 14
76131 Karlsruhe
Germany
e-mail: borissov@iism.uni-karlsruhe.de

Dirk Neumann
Chair for Information Systems Research
Albert-Ludwigs-Universität Freiburg
Kollegiengebäude II
Platz der Alten Synagoge
79085 Freiburg
Germany
e-mail: dirk.neumann@is.uni-freiburg.de

Mario Macías
Barcelona Supercomputing Center
Technical University of Catalonia (UPC)
c/ Jordi Girona, 29
08034 Barcelona
Spain
e-mail: mario.macias@bsc.es

Jordi Guitart
Barcelona Supercomputing Center
Technical University of Catalonia (UPC)
c/ Jordi Girona, 29
08034 Barcelona
Spain
e-mail: jordi.guitart@bsc.es

Jordi Torres
Barcelona Supercomputing Center
Technical University of Catalonia (UPC)
c/ Jordi Girona, 29
08034 Barcelona
Spain
e-mail: jordi.torres@bsc.es

Economic Models and Algorithms for Distributed Systems, 143–159
Book Series: Autonomic Systems

Mitigating Provider Uncertainty in Service Provision Contracts

Chris Smith and Aad van Moorsel

Abstract. Uncertainty is an inherent property of open, distributed and multi-party systems. The viability of the mutually beneficial relationships which motivate these systems relies on rational decision-making by each constituent party under uncertainty. Service provision in distributed systems is one such relationship. Uncertainty is experienced by the service provider in his ability to deliver a service with selected quality level guarantees due to inherent non-determinism, such as load fluctuations and hardware failures. Statistical estimators utilized to model this non-determinism introduce additional uncertainty through sampling error. Inability of the provider to accurately model and analyze uncertainty in the quality level guarantees can result in the formation of sub-optimal service provision contracts. Emblematic consequences include loss of revenue, inefficient resource utilization and erosion of reputation and consumer trust. We propose a utility model for contract-based service provision to provide a systematic approach to optimal service provision contract formation under uncertainty. Performance prediction methods to enable the derivation of statistical estimators for quality level are introduced, with analysis of their resultant accuracy and cost.

Mathematics Subject Classification (2000). Primary 91A40, 68M14, 68T99; Secondary 91A10.

Keywords. Grid computing, virtual organization, self organization, cooperative game theory

Supported in part by EPSRC grant EP/C009797/1 "Dynamic Operating Policies for Commercial Hosting Environment", EPSRC Grant EP/F066937/1 "Economics-inspired Instant Trust Mechanisms for the Service Provision Industry" and EU Network of Excellence grant 026764 "Resilience for Survivability in IST".

1. Introduction and motivation

Uncertainty is a central characteristic of domains where parties cease to retain complete control over outcomes in which they express preference. Open, distributed and multi-party systems demonstrate uncertainty with salience; with outcomes in such systems subject to non-determinism resulting from the inherently stochastic nature of a process, and to the lack of information pertaining to the process. The economic viability of the mutually beneficial relationships which motivate these systems is reliant on the ability of each party to model and analyze uncertainty in the outcomes of the relationships. This enables the establishment of expectations of each potential outcome of the relationship, and a rational ordering over relationships to be formulated.

Grid systems are pertinent examples of open, distributed and multi-party systems, uniting disparate resources from varied autonomous domains in a single large-scale system. A fundamental and motivating relationship in Grid systems is that of *service provision*, in which a *provider* supplies computational or storage *resources*, virtualized through a *service*, for use by a *consumer*. The consumer may consume the service directly, or may act as a downstream provider, reselling the service supplied with some *value-added* attributes. A quality of service (QoS) is associated with the provision, stipulating a set of quality level guarantees pertaining to the availability and performance of the service. These guarantees, along with payment terms for their fulfilment or violation, are formalized in a bilateral *service provision contract* between provider and consumer, established prior to service usage. The formation of the contract is habitually carried out by the provider, and requires the identification of those quality level guarantees which most closely represent business objectives.

Quality levels are, though, subject to uncertainty, attributable to inherent non-determinism in the service execution environment (see [11]), such as fluctuations in load or hardware and software failures. Uncertainty is additionally experienced in the modelling of the non-determinism using estimators based on empirical data sampled by an intrusive performance prediction method. The quality level guarantees included in the contract should, therefore, not only reflect business objectives, but should additionally consider the uncertainty in the contract perceived to fulfil these business objectives. Inability of the provider to effectively model and analyze this uncertainty in the formulation of service provision contracts can lead to the proposal of sub-optimal contracts, inconsistent with business objectives. Emblematic consequences of such errors include loss of revenue, inefficient resource utilization and erosion of reputation and consumer trust.

The provider requires a methodology for modelling uncertainty in quality levels, and for analyzing this uncertainty during the contract formation process, avoiding the use of ad hoc, informal and inaccurate methodologies. This ensures the optimality of the quality level guarantees under uncertainty, in the context of the business objectives. We address this need by extending common economic models to yield a utility model for contract-based service provision, providing a

systematic approach to service provision contract formation under uncertainty. Uncertainty in quality levels is modelled through the use of continuous random variables with some given probability distribution. This enables the expectation of any quality level to be established and incorporated into the formation of the service provision contract. We couple this utility model with performance prediction methods to enable the derivation of statistical estimators for quality level from empirical data, with analysis of the resultant accuracy and cost of these policies. The practical application of our utility model and performance prediction methods is demonstrated through their incorporation into a lightweight management architecture based on Representational State Transfer.

The remainder of this paper is structured as follows. We discuss previous work pertaining to uncertainty in service provision contracts in Section 2.. We then present our utility model for contract-based service provision in Section 3., and utilize this model in Section 4. to demonstrate the negative consequences of inaccuracy in statistical estimators for quality level. Performance prediction methods are discussed in Section 5., and results relating to the accuracy and cost of these methods are presented in Section 6.. A lightweight management architecture to support the utility model and performance prediction methods is discussed in Section 7., and we conclude and briefly summarize our work in Section 8..

2. Related work

Decision making under uncertainty is a well studied area in economics, psychology and management (see [1, 14]). The application of formalisms from these disciplines, particularly economics, to computing services is relatively novel. Service Level Agreements (SLAs) have been the focus of much of the work relating to computing services, with the aim of establishing a sound, viable economic basis for paradigms such as Grid [6] and Utility [10] Computing. Previous work has sought to mitigate provider uncertainty in SLAs through the control of system parameters [5, 9] in order to retain a given quality level. Other work has focused on consumer uncertainty in the willingness and ability of the provider to achieve a given quality level [2–4, 8]. This work, utilizing game theoretic concepts [7], assumes both that the provider has accurate estimators of uncertainty in quality levels, and that violation of the quality level by the provider is the result of defection and un-trustworthiness, rather than inherent non-determinism in the execution of the job.

We utilize the term Service Provision Contract (SPC) in place of Service Level Agreement (SLA) to emphasize verifiability and enforceability of the guarantees, and to emphasize a focus on computational and storage services rather than network services. Instead of system parameter control to retain a given quality level, we choose to enable the adaptation of quality level in accordance with system state. In contrast to other approaches, the values of a metric in our work are defined over a continuum rather than as discrete service levels, for example: *gold*, *silver*, and

bronze. This gives the potential to adapt the service level at a finer granularity in response to changes in system state. Additionally, we provide a basis not only for reasoning over contract formation, but also for the quantification of uncertainty in quality levels through the use of statistical estimators. The contract structure we utilize facilitates negotiation over quality level rather than uncertainty in quality level [4,8], which we consider a more natural approach whilst retaining the ability to reason over uncertainty in the value of a metric using probabilities.

3. Utility model for contract-based service provision

In this section, we extend common economic utility models to define a utility model for contract-based service provision. We show how the service provider can utilize this utility model to formulate optimal service provision contracts under uncertainty in quality levels.

Definition 3.1. A *quality level* is a pair $q = (m, v)$, where m is some chosen quality metric (such as response time) and $v \in \mathbb{R}$ is an assigned value for m (such as 300 milliseconds as bound for response time).

The service provision contract takes the form of a contingent contract, with payments conditional on a given outcome from the relationship. Without loss of generality, we define a contract to contain a single quality level guarantee, and define a binary outcome set for the relationship:

$$Q = \{q, \neg q\}\,. \tag{3.1}$$

Accordingly, the outcomes q and $\neg q$ represent the fulfilment and violation of the quality level guarantee respectively.

Definition 3.2. A *service provision contract* is a triple $a = (q, g_m(v), h_m(v))$. q is the quality level guarantee, $g_m(v)$ is the *ex ante* payment to the provider for provision of the service with quality q, and $h_m(v)$ is the *ex post* contingent payment to the consumer, payable on violation of the quality level guarantee q.

Contractual negotiation takes the form of a simple acceptance/rejection protocol. On request from the consumer, a contract, a, is formulated by the provider for the service provision and communicated to the consumer. The consumer then analyzes the contract to ensure consistency with his preferences. It is assumed the consumer has a given utility function expressing preferences over quality levels for the service, and that a given contract should fulfil a requirement threshold for the quality level, known as *reservation utility*. Failure to fulfil the threshold results in the rejection of the contract by the consumer and the termination of the protocol. On fulfilment of the threshold, the contract is accepted by the consumer, with confirmation sent to the provider who prepares the service for usage.

Let V_m denote a continuous random variable modelling the value, v, for a given metric, m. This random variable represents the inherent uncertainty in the value of metric m. We assume that such uncertainty is inherent due to the potential scale of system, and the inability to identify and quantify the complete set of influencing factors on resources. The probability density function for V_m is denoted $f_m(v)$, defined in the interval $[v_{min}, v_{max}]$. Let the ex ante payment for quality level q be defined by the function $g_m(v)$, and the ex post payment be defined by $h_m(v)$. For a given contract, we yield the expected utility function, $EU_m(v)$, in Equation 3.2.

$$EU(a) = \left(\int_{v_{min}}^{v} f_m(v)dv \cdot u\big(g_m(v)\big) \right) + \left(\int_{v}^{v_{max}} f_m(v)dv \cdot u\big(g_m(v) - h_m(v)\big) \right) . \tag{3.2}$$

The outcomes of the relationship are represented in Equation 3.2 by the two distinct terms. The fulfilment of the relationship, outcome q, is represented by the first term, where the bounds of integration determine the range of values over which the quality level is fulfilled. The violation of the relationship, outcome, $\neg q$, is represented by the second term, where the bounds of integration determine the range of values over which the quality level is violated. Dependant on how a given metric is appropriately constrained, the bounds of integration will change. For instance, if m is a *temporal* metric, such as response time, then it is appropriately constrained by $\leq$, and the integration bounds for outcome $\neg q$ would be v and v_{max}. Conversely, a *throughput* metric, such as data transfer per time unit, is appropriately constrained by $\geq$, and the integration bounds for outcome $\neg q$ would be v_{min} and v.

The functions $g_m(v)$ and $h_m(v)$ reflect the business objectives of the provider, defining the payment terms for a given quality level guarantee, q. Uncertainty is modelled through the incorporation of the density function, $f_m(v)$. $EU_m(v)$ consequently enables reasoning over these business objectives with explicit consideration of uncertainty. Given any value, v, of metric m, the expected utility can be calculated utilizing Equation 3.2. Accordingly, the optimal service provision contract can will utilize the optimal quality level, q^*, where:

$$\frac{dEU_m(v)}{dv} = 0\,. \tag{3.3}$$

The provider is assumed to be *risk-neutral*, and have an elementary utility function of the form:

$$u(x) = x\,. \tag{3.4}$$

The marginal utility of each additional unit of payment will therefore be constant, and the expected value and expected utility from a contract will be analogous. The objective of the provider is assumed to be the maximization of

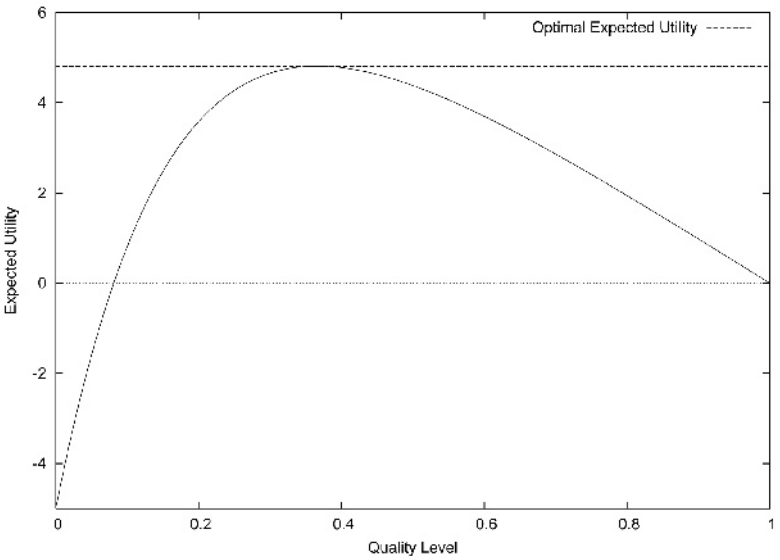

FIGURE 1. Quality level vs expected utility.
$V_m \sim Exp(5), g_m(v) = -10 \cdot v + 10, h_m(v) = -15 \cdot v + 15$

revenue, but other valid objectives such as minimization of losses or fulfilment of a given threshold for revenue could be easily represented in the model.

Figure 1 illustrates the establishment of the optimal quality level, q^*, for selected payment functions and quality level uncertainty. Different payment functions and density functions will yield differing optimal quality levels and differing expected utilities. The structure of the payment functions is discussed in more detail in Section 6..

4. Negative consequences of inaccurate quality level estimators

In this section, we illustrate the negative consequences of utilizing inaccurate statistical estimators for quality level uncertainty during the contract formation process.

Contract formation under uncertainty is reliant on the derivation of an estimator for $f_m(v)$. Given the potential scale and complexity of a Grid system, and the multitude of potential contributory factors to uncertainty in resources, $f_m(v)$ must be estimated through sampling of empirical data by an intrusive performance prediction method. Let the continuous random variable $\overline{V_m}$ denote the estimator of V_m derived from sampling, and let $\overline{f_m}(v)$ denote the estimator of the probability density function associated with this random variable. The smaller the error in the estimator $\overline{V_m}$, the more accurate the contract formation process, and the

closer the yielded contract will be to the optimal under perfect information.

$$\overline{EU}(a) = \left(\int_{v_{min}}^{v} \overline{f_m}(v) dv \cdot u\big(g_m(v)\big) \right) + \left(\int_{v}^{v_{max}} \overline{f_m}(v) dv \cdot u\big(g_m(v) - h_m(v)\big) \right) . \tag{4.1}$$

Optimal service provision contracts will be yielded when $\overline{f_m}(v) = f_m(v)$. This represents perfect information pertaining to quality level uncertainty. We refer to the expected utility, given this perfect information, as the Expected Value of Perfect Information (EVPI) (see [1]). The more imperfect the information on which we base our contract formation, the greater the inaccuracy of our estimator. This inaccuracy increases the risk of sub-optimal contract formation, which can take two distinct forms.

Firstly, the contract may be sub-optimal due to over-estimation of quality level guarantees. The imperfect information on which our estimator is based is giving the provider false confidence in the ability to deliver a given quality level. Over-estimation, therefore, leaves the provider at risk from quality level violations, and consequently an increase in the frequency of ex post payments, and an erosion of trust and reputation in the willingness and ability of the provider to fulfil the guarantees. This issue of trust and reputation is particularly damaging, since both are valuable and highly volatile properties in domains of self-interested, autonomous parties such as Grid systems.

Secondly, the contract may be sub-optimal due to under-estimation of quality level guarantees. The imperfect information on which our estimator is based is leading the provider to be overly conservative in his ability to deliver a given quality level. Under-estimation leaves the provider vulnerable to lost revenue, attributable to under-utilization of resources and thus organizational slack. A key motivation for Grid and Utility computing is high-utilization of resources and the ability to dynamically adapt the policies which control resource usage to achieve this. Under-estimation of the ability to deliver a given quality level hinders this motivation through inaccurate portrayal of system state.

Inaccurate estimators are therefore damaging to the contract formation process, causing both tangible (payment) and intangible (trust/reputation) losses to the provider and affecting the economic viability of the service provision relationship. Figure 2 illustrates the effect of inaccurate estimators on the expected utility of a given contract. The opportunity loss illustrates the extent of the sub-optimality for given levels of inaccuracy in the estimators. For illustrative purposes, the error was introduced to the estimator of the rate parameter, λ, of the exponential distribution. Section 5. discusses the different methods by which empirical data can be utilized to derive such an estimator.

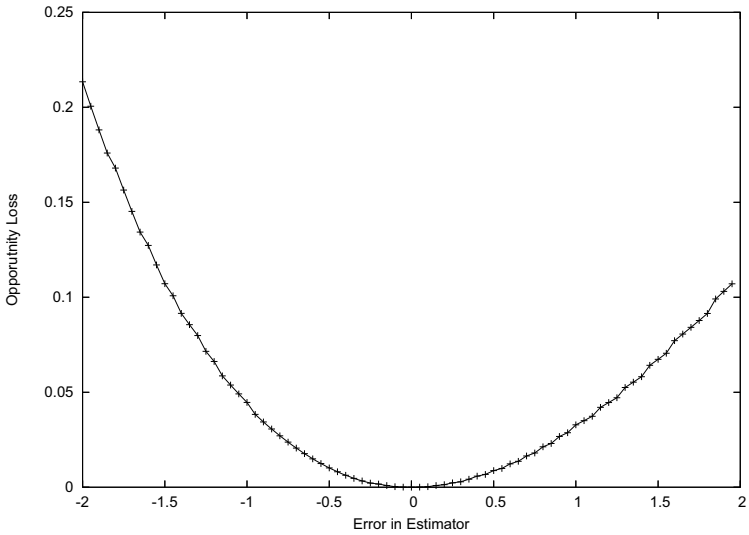

FIGURE 2. Error in estimator vs opportunity loss.
$V_m \sim Exp(5), g_m(v) = -10 \cdot v + 10, h_m(v) = -15 \cdot v + 15$

5. Performance prediction methods for derivation of quality level estimators

This section addresses the need for statistical estimators of quality level uncertainty, given the negative consequences of inaccuracies outlined in Section 4.. We discuss the effect of different performance prediction methods to derive an estimator, utilizing varied sampling strategies of empirical data. We address the notion of perfect information, and the trade-off between cost and accuracy which must be resolved when striving for this perfect information.

The process of performance prediction is assumed to be intrusive, in the sense that resources utilized for performance prediction could be otherwise utilized. That is, there exists an *opportunity cost* for performance prediction. This cost is heavily dependant on the scale of resources demanded by the performance prediction method. The scale of resources is determined by the sampling strategy (see [13]), which includes the sample size and sample selection criterion. With each sampling strategy there exists not only a cost, but a benefit in terms of information. We utilize the term Expected Value of Sample Information (EVSI) (see [1]) to denote this benefit. The EVSI represents the benefit gained from the sampling strategy and clearly when information comes at a cost we only wish to incur that cost if appropriate benefits are accrued.

Definition 5.1. A *performance prediction method*, p, is a sampling strategy utilizing a sample set of empirical data with a given cardinality.

We define a set of n different performance prediction methods, $p \in P$, for a given system:

$$P = \{1, \ldots, n\}\,. \tag{5.1}$$

Let $\overline{V_m^p}$ denote a continuous random variable representing the value v of a given metric m, using performance prediction method p where $p_1 \leq p \leq p_n$. Additionally, for each $\overline{V_m^p}$ let the estimator of the probability density function obtained through performance prediction be denoted $\overline{f_m^p}(v)$. For a given performance prediction method p, we associate a cost, denoted by $k(p)$. Equation 3.2 is subsequently amended to include the estimators and the cost for the performance prediction method yielding Equation 4.1.

$$\begin{aligned}\overline{EU}(a,p) = &\left(\int_{v_{min}}^{v} \overline{f_m^p}(v)dv \cdot u\big(g_m(v)\big)\right) \\ &+ \left(\int_{v}^{v_{max}} \overline{f_m^p}(v)dv \cdot u\big(g_m(v) - h_m(v)\big)\right) \\ &- k(p)\,.\end{aligned} \tag{5.2}$$

Given a performance prediction method, and the resultant sample set of empirical data, the probability density function $\overline{f_m^p}(v)$ is required. Two distinct options exist for the derivation of this density function. Firstly, the sample data could be utilized to bootstrap a parametric distribution such as a normal or exponential distribution. This option requires *a priori* knowledge of the specific distribution of the values of given metric, and thus the parameters which should be derived. Secondly, the sample data could be utilized to derive an empirical distribution, which could itself be used or could be fitted to a parametric distribution. This options requires no *a priori* knowledge of the distribution of metric values. Indeed, the significant assumption of *a priori* knowledge of the distribution of the metric decreases the value of information yielded from a performance prediction method. The information, in the case of a parametric distribution, carries less value as the class of distribution is known, and only requires refinement through derivation of appropriate parameter estimators.

The sampling cost, $k(p)$, associated with each performance prediction method is assumed to increase with p. This increase in cost represents increasingly intrusive resource usage by the performance prediction method. Conversely, error in the estimator resulting from the performance prediction method will, by the law of large numbers, decrease with p. This reduced error represents a movement towards perfect information on which to base decisions. In selection of a performance prediction method, we must therefore resolve the trade-off between $k(p)$ and the $EVSI$ for p. The resolution point of this trade-off is heavily dependant on the

form of $k(p)$ and the tolerance of the decision making logic (Equation 5.2) to errors in the estimator. Clearly, in the case of the provider, the payment functions and the utility function are primary influencing factors in the decision making logic, and accordingly dictate the tolerance to errors in the estimator. In cases where little tolerance to error exists, the $EVSI$ will be high, and accordingly a higher value of $k(p)$ can be justified, and vice versa.

6. Results

In this section, we present the results of simulation work utilizing our utility model (Section 3.) in conjunction with varied performance prediction methods (Section 5.).

The performance prediction methods utilized in this simulation work are defined by a set P (Equation 5.1), where $n = 20$. For each policy, p, we defined the cardinality of the sample data set using a linear function:

$$|S_p| = \alpha \cdot p\,. \tag{6.1}$$

In addition, we define a cost for a given performance prediction method using the function given in Equation 6.2. The β term enables the representation of a variable cost for performance prediction, increasing with the size of the sample data set. For example, the cost of analyzing the data, in terms of resource usage, would be variable on the size of the data set. The γ term enables the representation of a fixed cost for performance prediction, such as investment in a performance prediction infrastructure.

$$k(p) = \beta \cdot p + \gamma\,. \tag{6.2}$$

The experimental procedure utilized an exponential density function, $f_m(v)$, to represent perfect information pertaining to quality level uncertainty. The exponential distribution was chosen to be indicative of the values of a metric such as response or service time.

$$f_m(v) = \lambda \cdot e^{-\lambda \cdot v}\,. \tag{6.3}$$

Given the exponential density function, each performance prediction method derived an estimator, $\overline{f_m^p}(v)$, for $f_m(v)$. This estimator was derived through the formation of an empirical distribution from sampling (as defined by 6.1) of the perfect information, $f_m(v)$. The resultant estimator, $\overline{f_m^p}(v)$, was then be incorporated into Equation 5.2 along with the appropriate cost function (Equation 6.2), and payments functions:

$$g_m(v) = \delta_g \cdot v + \epsilon_g \tag{6.4}$$

$$h_m(v) = \delta_h \cdot v + \epsilon_h\,. \tag{6.5}$$

Each metric utilized in an SPC is defined over a given range of values [v_{min}, v_{max}]. In order to simplify the definition of the payment functions, and the enable

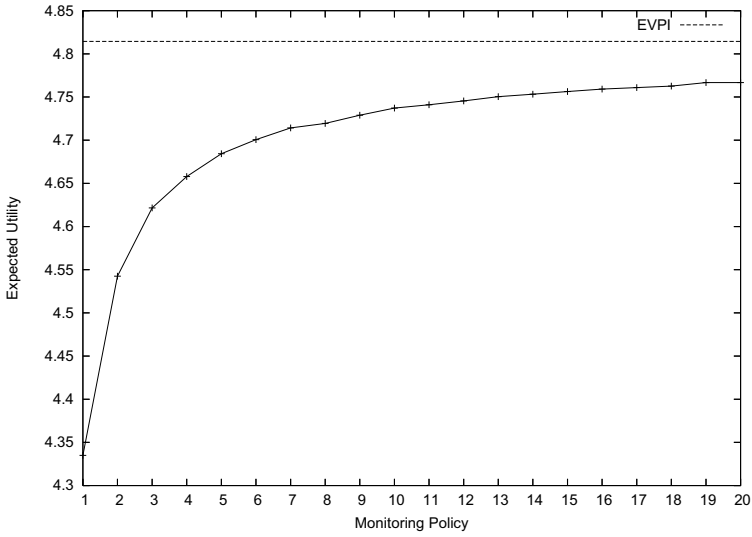

FIGURE 3. Performance prediction method vs expected utility.
$\alpha = 10, \beta = \gamma = 0, \lambda = 5, \delta_g = -10, \epsilon_g = 10, \delta_h = -15, \epsilon_h = 15$

the potential reuse of payment functions over multiple metrics, we normalize the values of any given metric to be defined over the range of values $[0, 1]$ using:

$$v' = \frac{v - v_{min}}{v_{max} - v_{min}}\,. \tag{6.6}$$

Through manipulation of the coefficients: δ_g and δ_h, and the constant terms: ϵ_g and ϵ_h, the payments can be made increasingly or decreasingly variable on the quality level. This enables the straightforward representation of a wide range of payment functions. Clearly, the selection of a payment function will significantly effect the contract formation process under uncertainty. A study of the impact of differing payment functions within the contract formation process is beyond the scope of this paper. For each performance prediction method, the expected utility from the contract yielded was compared with that yielded under perfect information.

Figure 3 illustrates the effect on expected utility of each performance prediction method. EVPI denotes the expected utility from perfect information pertaining to the quality level uncertainty, $\overline{f_m^p}(v) = f_m(v)$. As the sample data set utilized by the performance prediction method increases in cardinality, the ex-

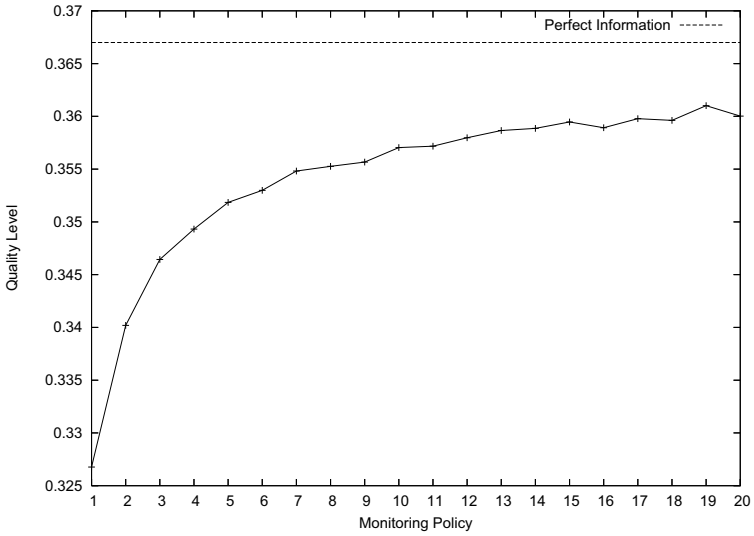

FIGURE 4. Performance prediction method vs quality level.
$\alpha = 10, \beta = \gamma = 0, \lambda = 5, \delta_g = -10, \epsilon_g = 10, \delta_h = -15, \epsilon_h = 15$

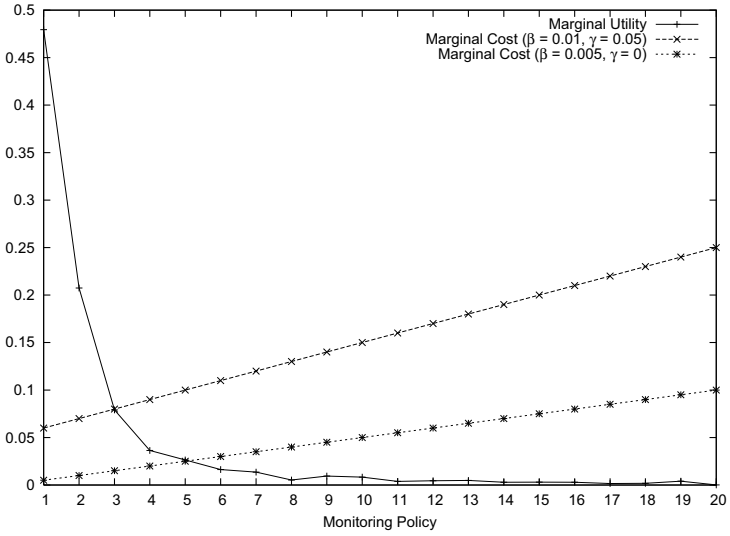

FIGURE 5. Performance prediction method vs marginal utility/marginal cost.
$\alpha = 10, \lambda = 5, \delta_g = -10, \epsilon_g = 10, \delta_h = -15, \epsilon_h = 15$

pected utility derived from the contract increases and, by the law of large numbers, $\overline{f_m^p}(v) \to f_m(v)$ as $p \to \infty$. The marginal increase in expected utility (marginal utility), though, decreases as the sample set size increases in cardinality. The EVSI for the less intense performance prediction methods is therefore greater than that of the more intense policies, contributing an increasing amount to expected utility. The decreasing EVSI of more intense performance prediction methods is additionally demonstrated by Figure 4. As we increase the cardinality of the sample data set, the quality level we quote in our contract approaches the optimal, and again, in the limit would reach this optimal as $p \to \infty$.

The decrease in EVSI as the performance prediction method becomes more intense highlights an important consideration with regard to performance prediction cost. Clearly, we only wish to incur a cost for performance prediction, if by doing so we accrue equal or greater benefit in terms of expected utility. We must therefore find the optimal performance prediction, where the cost of performance prediction and the benefit of performance prediction are analogous. Figure 5 illustrates such optimal policies for two different performance prediction cost functions. The first monitoring cost function ($\beta = 0.01, \gamma = 0.05$) dictates that $p = 3$ is the optimal performance prediction method, given the defined payment and cost functions, whilst the second function ($\beta = 0.005, \gamma = 0$) dictates that $p = 5$ is the optimal performance prediction method. The provider should therefore establish the optimal performance prediction policy based on the cost of each performance prediction method and the marginal utility, or EVSI, that each performance prediction policy provides. This ensures the optimal mitigation of uncertainty in the contract formation process.

7. Implementation

In this section, we present an implementation of our utility model and performance prediction methods, utilizing a lightweight management architecture based on Representational State Transfer (REST) [12].

The architecture comprises of four principal components, as shown in Figure 6. The *contract manager* is the entry point for all requests to the system. The component is responsible for any logging and enactment required by these requests. A key task of the contract manager is the control of the contract life-cycle, from creation to enactment to termination or expiry. Any requests for jobs in accordance with a given SPC are checked for validity by the contract manager, and any payments due to or from the provider as part of a given SPC are administered by the contract manager.

The creation of an SPC is the task of the *contract factory*. On reception of a contract request, the contract manager invokes the contract factory component to create an appropriate contract. The invocation of the contract factory requires the definition of appropriate payment functions and an appropriate utility function.

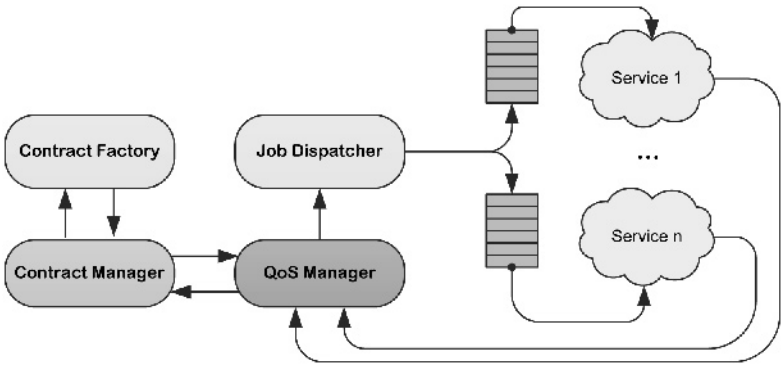

FIGURE 6. Management architecture diagram.

We assume both functions are of the form:

$$p = a_n x^n + \cdots + a_1 x + a_0 \,. \tag{7.1}$$

Additionally, the contract factory requires the estimator of the uncertainty in the values for a given metric, such that the optimal contract can be formalized. Given the payment functions, the utility function, and the appropriate estimator, the optimal contract is formulated by the contract factory utilizing Algorithm 1.

Algorithm 1: Optimal contract formation algorithm.

input : Metric, m
input : Payment functions, $g_m(v)$ and $h_m(v)$
input : Probability density function, $\overline{f_m^p}(v)$
input : Performance prediction cost, $k(p)$
input : Utility function, $u(x)$
output: Optimal contract, a^*

$eu \leftarrow$ `EUBuild(`$g_m(v), h_m(v), \overline{f_m^p}(v), k(p), u(x)$`)`;
$dv \leftarrow$ `Differentiate(`eu, v`)`;
$v^* \leftarrow$ `BisectionMethod(`dv`)`;
$q^* \leftarrow (m, v^*)$;
$a^* \leftarrow (q^*, g_m(v^*), h_m(v^*))$;

The use of polynomial functions for the payment functions and the utility function enables the application of standard differentiation rules for polynomials to be applied, simplifying the optimization process. A bisection algorithm is used by the contract factory to find the optimal value for the metric after differentiation. This bisection algorithm was chosen for its simplicity and robustness, yet other

root-finding algorithms such as Newton's method or the Secant Method could be simply integrated.

The *QoS manager* component is responsible for the execution of the performance prediction method, and thus the sampling on which the estimator is based. Job requests pass through the component on both prior and post execution. In accordance with the selected performance prediction method, the estimator for a given metric is calculated from a given sample set of the relevant empirical data from job requests. The component enables the estimators for values uncertainty in the values of a metric to be represented by either an empirical distribution or a parametric distribution. Selection of the appropriate distribution is dependant on the level of domain knowledge available pertaining to a given metric and the values it may yield.

Once a job is ready for execution, the *dispatcher* component is invoked to place the job in the appropriate queue for the service. Any number of different services can be supported by the dispatcher, with a separate queue for each service. On availability of the required compute or storage resources, the next job in the queue for a given service is dispatched to that resource and executed. The completed job passes through QoS manager to enable the calculation of each metric value for the job, and the contract manager then enacts any required payment actions.

In accordance with the principles of REST, we model stateful entities as *resources*. Resources are used to represent the primary stateful components in the interaction between the consumer and the provider, that is, contracts and jobs. Each contract and job is modelled as a Finite State Automata (FSA). The resources expose these FSAs and enable introspection and manipulation of their state through some representation, resolvable at a uniquely assigned identifier (URI). The introspection and manipulation operations are executed using HTTP methods. The consumer is able to view and manipulate the state of both the contract and the jobs, as defined by the underlying FSA. On execution of these operations the consumer receives a HTTP response which contains a status code, denoting the status of the request. The consumer may additionally receive in this response a representation of the resource, the FSA. For instance, the consumer may wish to check the status of some contract. The contract has the URI: http://example/contract?12345, and thus at this URI a representation of the state of the contract can be found. To retrieve this representation the consumer executes a GET operation on http://example/contract?12345, receiving in response both a representation and a status code (for example *200 OK* on success). The representation of the contract contains not only the instance-specific information, it additionally contains any URIs required to manipulate the contents of this contract. This is an example of the elegance of stateless interactions, requiring no state to be held on the consumer-side relating to the contract state. The consumer can derive the state of the contract and the conceivable transitions from that state all from the representation of that state.

8. Conclusion

In this paper, we have presented a utility model for contract-based service provision which provides a systematic approach to the formation of optimal service provision contracts under uncertainty in quality levels. We have demonstrated the negative consequences of inaccurate quality level estimators and the sub-optimal contracts they yield. A variety of performance prediction methods have been presented to facilitate the derivation of quality level estimators whilst addressing a trade-off between the accuracy of these estimators and the cost of their derivation. Finally, an implementation of the utility model and performance prediction methods was described, utilizing an architecture based on Representational State Transfer.

References

[1] B. F. Baird. *Managerial Decisions under Uncertainty*. Wiley Series in Engineering & Technology Management. Wiley and Sons, Inc, 1989.

[2] H. K. Bhargava and S. Sundaresan. Managing quality uncertainty through contingency pricing. In *HICSS'*03*: Proceedings of the* 36*th Annual Hawaii International Conference on System Sciences*, Washington, DC, USA, 2003. IEEE Computer Society.

[3] S. Brainov and T. Sandholm. Contracting with uncertain level of trust, 1999.

[4] A. Byde. Incentive-compatibility, individual-rationality and fairness for quality of service claims. Technical report, Hewlett Packard, 2006.

[5] Y. Chen, S. Iyer, X. Liu, D. Milojicic, and A. Sahai. Sla decomposition: Translating service level objectives to system level thresholds. Technical report, Hewlett Packard, January 2007.

[6] I. Foster and C. Kesselman. *The Grid* 2*: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.

[7] D. Fudenburg and J. Tirole. *Game Theory*. MIT Press, 1991.

[8] B. Huberman, F. Wu, and L. Zhang. Ensuring trust in one time exchanges: Solving the QoS problem. *Netnomics*, 7(1): 27–37, 2005.

[9] J. Palmer, I. Mitrani, M. Mazzucco, P. McKee, and M. Fisher. Optimizing revenue: Service provisioning systems with qos contracts. Technical report, School of Computing Science, Newcastle University, May 2007.

[10] M. A. Rappa. The utility business model and the future of computing services. *IBM Systems Journal*, 43(1), 2004.

[11] D. A. Reed, C. L. Mendes, C. da Lu, I. Foster, and C. Kesselman. *The Grid* 2*: Blueprint for a New Computing Infrastructure – Application Tuning and Adaptation*. Morgan Kaufman, San Francisco, CA, second edition, 2003. pp.513–532.

[12] C. Smith and A. van Moorsel. On efficient stateful resource management. Technical report, School of Computing Science, Newcastle University, May 2006.

[13] S. K. Thompson. *Sampling*. Wiley Series In Probability And Statistics. Wiley, 2002.

[14] A. Tversky and D. Kahneman. Judgement under uncertainty: Heuristics and biases. *Science*, 185(4157): 1124–1131, September 1974.

Chris Smith
School of Computing Science
Newcastle University
Newcastle-upon-Tyne
NE1 7RU
United Kingdom
e-mail: c.j.smith4@ncl.ac.uk

Aad van Moorsel
School of Computing Science
Newcastle University
Newcastle-upon-Tyne
NE1 7RU
United Kingdom
e-mail: aad.vanmoorsel@ncl.ac.uk

Economic Models and Algorithms for Distributed Systems, 161–180
Book Series: Autonomic Systems

Text-Content-Analysis based on the Syntactic Correlations between Ontologies

Axel Tenschert, Ioannis Kotsiopoulos and Bastian Koller

Abstract. The work presented in this chapter is concerned with the analysis of semantic knowledge structures, represented in the form of Ontologies, through which Service Level Agreements (SLAs) are enriched with new semantic data. The objective of the enrichment process is to enable SLA negotiation in a way that is much more convenient for a Service Users. For this purpose the deployment of an SLA-Management-System as well as the development of an analyzing procedure for Ontologies is required. This chapter will refer to the BREIN, the FinGrid and the LarKC projects. The analyzing procedure examines the syntactic correlations of several Ontologies whose focus lies in the field of mechanical engineering. A method of analyzing text and content is developed as part of this procedure. In order to so, we introduce a formalism as well as a method for understanding content. The analysis and methods are integrated to an SLA Management System which enables a Service User to interact with the system as a service by negotiating the user requests and including the semantic knowledge. Through negotiation between Service User and Service Provider the analysis procedure considers the user requests by extending the SLAs with semantic knowledge. Through this the economic use of an SLA-Management-System is increased by the enhancement of SLAs with semantic knowledge structures. The main focus of this chapter is the analyzing procedure, respectively the Text-Content-Analysis, which provides the mentioned semantic knowledge structures.

Mathematics Subject Classification (2000). Primary 91A40, 68M14, 68T99; Secondary 91A10.

Keywords. Uncertainty, contracts, service provision, distributed systems, performance prediction

1. Introduction

The aim of this chapter is to explore several Ontologies in order to analyze their content and extract the matching concepts in another Ontology. For this it is quite important to analyze the syntactic correlations of a set of Ontologies. To meet the requirements for these issues, a method for analyzing text and content within the Ontologies is needed. This method is called Text-Content-Analysis which will be investigated in this chapter. The results will be used to enhance SLAs and support the negotiation process in order to obtain a much more efficient SLA-Management-System.

However, the goal is not to build several new Ontologies for a new kind of language understanding but to analyze given standard Ontologies. Hence, the used standard Ontologies should be related to the research area of interest. The concern is to generate an understanding by analyzing several Ontologies and develop an overlapping understanding. For this process existing Ontologies will be determined. The focus of the selected Ontologies pertains to those in a selected scientific research field. However, it will be possible to use the Text-Content-Analysis for different research fields but for each analysis the topic of the Ontologies has to be clarified before the analysis is performed. The work described in this chapter is concerned with the selection of Ontologies related to the area of mechanical engineering in order to use the knowledge structures of the Ontologies to enrich the negotiation of Service Level Agreements.

It is the intention of this work to deploy a matching procedure for Ontologies in order to use the Text-Content-Analysis to enrich a new Ontology with the extracted results. The benefit of the analysis is the availability of a wider range of semantic knowledge within one new Ontology. This new Ontology can be used to extend Service Level Agreements by semantic knowledge structures. In this case, we will determine Ontologies from the field of mechanical engineering. For the analysis it is important to get a variety of Ontologies whose alignment lies in the field of engineering in order to ensure that the focused Text-Content-Analysis generates useful semantic knowledge structures. The proposed method of analyzing Ontologies will be related to existing methods.

Another basic module which is presented in this chapter will be an SLA-Management-System whose intention is to facilitate negotiation of SLAs. Based on the negotiated SLAs the analysis of the syntactic correlations takes place. To be more preceise, this means that we will investigate syntactic correlations within a set of Ontologies which fit to the SLAs. The SLA consists of certain terms which we will try to understand in a more expressive way by comparing the SLA terms with the given syntactic correlations. The procedure will be deployed as a service which allows the interaction between a Service User and Service Provider. The deployment of Text-Content-Analysis (TCA) as a service will demonstrate the benefit of combining the TCA and the SLA-Management-System. Users' requests can be negotiated by the SLA-Management-System. Hence, it becomes possible to adapt additional semantic knowledge to SLAs by using data structures which

are extracted through the Text-Content-Analysis. This method includes a service component which is responsible for the interaction with the user. This makes the method more convenient to the requirements of the user. The described method interacts in a dynamic way by the use of the service component which supports the interaction between user and the SLA-Management-System. In the following sections, we will discuss the different steps and the scope of the focused Text-Content-Analysis by exploring syntactic correlations between Ontologies. Furthermore, the functionality of the SLA-Management-System and the adoption of the semantic knowledge structures which are extracted by the Text-Content-Analysis will be discussed in more detail.

2. Description of work

The best way to describe our work, is to align it to the workflow of the procedure for analyzing Ontologies and the resulting enhancements of the SLA-Management-System. With respect to this, many steps need to be taken, therefore it is of high importance to organize them precisely. First of all it is necessary to evaluate carefully those Ontologies. In order to ensure an adequate level for the Text-Content-Analysis there are some questions which have to be answered.

- Are there standard Ontologies available for the research field to be analyzed?
- Are the selected Ontologies efficient enough to extract semantic knowledge structures out of them?

Therefore it has to be ensured that the the selection of used Ontologies fits with the terms of the SLA so that it is possible to use the data structures within the Ontologies. Otherwise, there won't be any benefit of analyzing the set of Ontologies because there is no accordance between Ontology content and SLA terms. Ontologies for the aspirated use case are those from the field of technical engineering. Therefore, we need workable Ontologies available fulfilling the needed requirements. Furthermore, the selected Ontologies should support common standards such as RDF-(S) and OWL in order to address the Text-Content-Analysis of a wide range of users. For this the open source community is in the focus of the work described in this chapter and therefore it is a prerequisite to use open source standards. Hence, the free open source platform Protégé is an adequate tool which seems to be beneficial to the indicated issues.

In terms of evaluating the Ontologies it is essential to develop a matching algorithm. The implementation of this algorithm is a principal component of the targeted work and the whole Text-Content-Analysis. By this, the matching between the set of Ontologies takes place. Therefore the matching algorithm is of vital interest for the presented work and the Text-Content-Analysis. After the matching of the given set of Ontologies is guaranteed, the Text-Content-Analysis can be developed. Within the analysis the matching algorithm is taken to extract the results out of the Ontologies which are required to generate a new Ontology

containing the results of the matching procedure. When the matching algorithm and the Text-Content-Analysis are developed the SLA-Management-System has to be installed. Through this, it becomes possible to apply the semantic data structures to the SLAs which are negotiated in the SLA-Management-System. The available semantic knowledge supports the negotiation process through making SLA terms more expressive to a user, e.g. by adding the extracted results from the analysis of the selected Ontologies. Therefore, the economic use of SLAs is supported by providing the user with additional semantic knowledge related to the SLA terms. Through the service oriented aspect of the work within this chapter a user will be enabled to use the semantic knowledge structures for his needs in a extensible manner. The Service User has the option of selecting Ontologies, to start the Text-Content-Analysis and to utilize the results of the Text-Content-Analysis for SLA negotiation. In terms of economic usage this means that the negotiation of SLAs becomes much more convenient to the user, as SLAs will be usable also for business entities, who are no experts and have only limited knowledge about the SLA terms.

The workflow will be organized by the following steps:

- manual selection of applicable Ontologies
- investigation of methods for analyzing Ontologies
- development of a matching algorithm related to the previous step
- modeling the Text-Content-Analysis based on the developed matching algorithm
- developing the Text-Content-Analyzing method
- installing an adequate SLA-Management-System
- relating semantic knowledge structures with the SLA-Management-System

By involving the SLA-Management-System to the whole structure the work obtains a service oriented aspect as the SLA-Management-System and the Text-Content-Analysis are used as services.

2.1 Analyzing ontologies

For the presented work within this chapter Ontologies are an adequate resource for extracting semantic knowledge out of a given set of Ontologies by investigation of the given concepts and relations. These are issues which are (at least partially) addressed by the European project BREIN. BREIN includes aspects such as knowledge representations and how to extract required data out of them. In the BREIN project RDF-(S) and OWL are used for knowledge presentation.

On the one hand, when thinking about knowledge representation and technologies of the semantic web such as RDF-(S) and OWL, the European LarKC project should be considered. One main aspect of the LarKC project is about semantic reasoning by using knowledge bases and Ontologies. The general idea of LarKC is to develop a platform which is able to combine and execute several plug-ins. The plug-ins are used in the platform ("plugged in") in order to deploy a

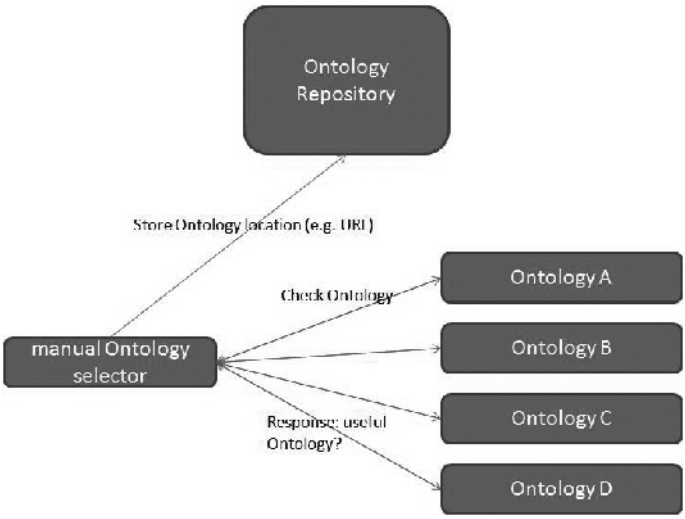

FIGURE 1. Selection of ontologies.

pipeline (consisting of several plug-ins) which ensures the processing of data structures such as SPARQL queries. Through this an extensible architecture is given which allows reasoning within semantic knowledge bases. Furthermore, including plug-ins allows usage bya a wide range of developers which can create their own modules ("plug-ins") and add them to the platform.

All these activities address the field of semantic web, which thereby shows its high importance. Semantic Web technologies and tools provide a systematic method in explicit knowledge capture and dissemination. Ontologies are describing an accumulation of concepts and the relationships between the concepts. By the use of Ontologies semantic knowledge structures are provided which are utilizable for the requirements of this work. However, it is necessary to make the semantic knowledge structures explicit and to search inside the Ontologies in order to match concepts of several Ontologies. Therefore, the syntactic correlations between the Ontologies will be exposed to merge the given set of Ontologies for understanding the content in a way that is helpful for further work. The analysis of the Ontologies will take place through a matching process within the Ontologies. But before these steps will take place it is essential to evaluate which Ontologies can be used for the envisaged purpose.

As mentioned before, the procedure is based on Ontologies whose scope is technical engineering. A first step of the work is to evaluate useful Ontologies and to store their location (e.g. the URLs) in some kind of repository. There are several aspects that have to be clarified such as the range of specifications that have to be supported in order to integrate the selected Ontologies (Figure 1).

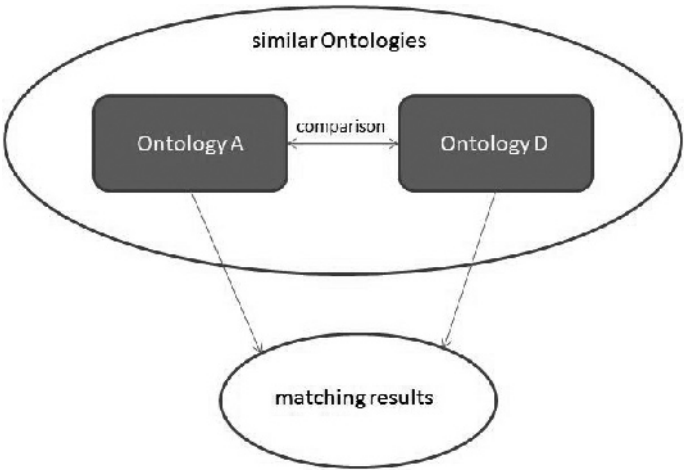

FIGURE 2. Matching between ontologies.

In order to use the method widely, open source community standards such as RDF-(S) and OWL will be applied. Thus the selected Ontologies have to support these specifications. Another issue is to evaluate useful Ontologies for the described work. If an Ontology is based on RDF-(S) or OWL and lies in the focused scope it might be relevant. However, the aim of the work presented in this chapter is to develop a matching algorithm for the Text-Content-Analysis which is not designated to a specific research field, but should be applicable to every focused research which provides Ontologies related to the SLA terms that should be negotiated in the SLA-Management-System.

After defining and evaluating the set of Ontologies the next step is to explore a matching method for analyzing Ontologies which comply with the given requirements. To achieve this method, existing Ontology-analysis will be considered in order to extend already existing methods. Through this a new improved analysis method which is based on proven techniques is developed and a new ontology-analysis which is adapted to the given requirements of a user is generated (Figure 2).

Therefore, Ontologies must be evaluated with the aim of meeting the users' requirements. Furthermore, a new analysis based on the chosen Ontologies and existing analysis of Ontologies has to be developed. Hereby, the next step of the work can be initiated.

2.2 Text-Content-Analysis

In order to analyze the textual content of the given Ontologies in an adequate manner the Text-Content-Analysis, which refers to the matching algorithm, is developed. This development is based on the evaluation and the analysis of On-

tologies. In the following we will take a closer look at the matching of the selected Ontologies and afterwards how this matching is used for Text-Content-Analysis.

2.2.1 Matching algorithm

In order to ensure a high quality extraction of semantic knowledge it is obvious to develop a robust and reliable matching algorithm meeting the requirements of the user. Therefore, the first step is to determine exactly which requirements we are talking about. As mentioned before the main goal of the presented work is to ensure a matching of several semantic structures in order to achieve a new knowledge base in terms of an Ontology, or more specifically one new Ontology which includes the concepts of the underlying Ontologies analyzed before. Through this we can assume that the requirements are on the one hand receiving knowledge structures which are in the focus of the related work, and on the other hand the possibility to use only one Ontology which deals with the extracted knowledge out of the underling Ontologies. Regarding the first assumption which is related to selecting useful Ontologies, it is obvious that this is possible in the selection phase. By that it is ensured that the Ontologies are in the scope of the envisaged work described in this chapter. However, the second assumption is much more complex. The provisioning of one upper Ontology[1] which includes the extracted results out of the selected Ontologies presumes the mentioned matching algorithm. Therefore, we will take a closer look at the matching and the extraction of the knowledge structures.

Before starting we have to think about some facts when talking about matching Ontologies. First of all the matching algorithm refers to the given set of Ontologies which is composed by the user. At this stage a user with background knowledge about the focused work and a little knowledge about Ontologies is postulated because of the need to select useful Ontologies. The user will be supported when selecting Ontologies by the Text-Content-Analysis but he has to think himself about the sense of his selection. The idea is that similar Ontologies are selected with the aim of extracting a high number of results out of the given set of Ontologies. If not, the matching of the Ontologies could be performed as well but it won't make any sense to compare concepts from Ontologies which are completely different, e.g. comparing a biomedical Ontology with an Ontology about engineering will not produce useful results.

To keep it simple for the start we will restrict the matching method to two Ontologies, OntoA and OntoB. The algorithm searches through the Ontologies and compares the concepts within the Ontologies. When matching the concepts of both Ontologies there are three types of matching, a full matching between the concepts if the concepts are really identical, a non-matching if there is no matching concept within the compared Ontology and a partially matching if there is a

[1] Upper Ontology: A top-level Ontology which describes general concepts of a number of underlying Ontologies

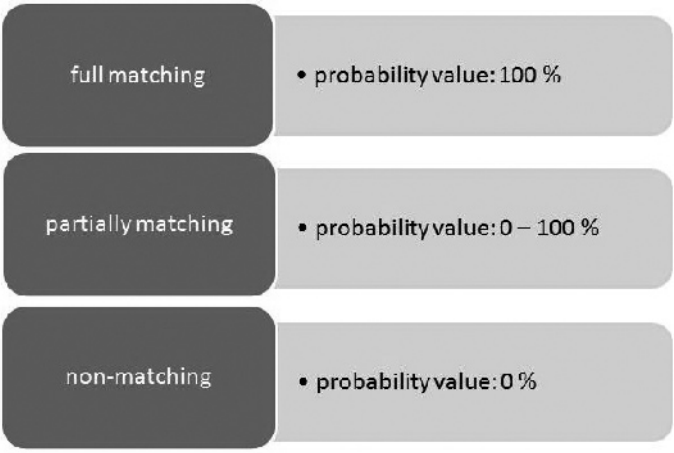

FIGURE 3. Matching types.

matching between two compared concepts which does only fit to a certain part of the concepts. To handle a full or a non-matching between concepts will be a well proofen but very simple method of doing comparisons. Hence, the approach of including partial matches extends the capabilities of matching concepts in a wide range. However, to ensure the quality of the extracted results out of the comparisons between the concepts an indicator which presents the degree of accordance is needed. For this a probability value is used to express the matching level between the concepts, which is added as property to all three types of matching (Figure 3).

As a result of this we will not distinguish between the mentioned types of matching but we will distinguish between the degrees of accordance. This method for generating the probability value ensures its accuracy and usability.

When thinking about the both example Ontologies OntoA and OntoB we can demonstrate how the probability value is generated. In the case that OntoA has a concept ConA and OntoB has a concept ConB there will be a matching between ConA and ConB. Now there are several possibilities. If there is no matching between ConA and ConB the probability value is 0 percent, if there is a full matching between ConA and ConB we the probability value is 100 percent and in the case that there is something between, respectively a partially matching, the probability value is calculated by the number of matching features in proportion to the whole number of features available (Figure 4).

When we think of the given example there will be one new concept which includes the extracted results out of both concepts ConA and ConB. Furthermore, the new concept is not the result of a 100 percent matching but a partially matching of both underlying concepts. In this case the new concept is *car/bicycle* which has one *isa* and one *can* connection and one probability value.

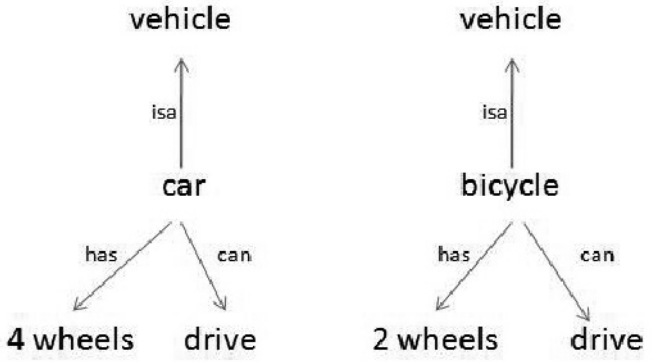

FIGURE 4. Generating the probability value.

- *isa:* vehicle
- *can:* drive
- *probability value:* X percent

Now we have to think about the formula for generating the probability value. In this case there are two concepts which are merged to one new concept in an upper Ontology. ConA has three relations (*isa, has, can*) and two of them are matching. By this there is a matching probability for ConA of 66,6 percent (rounded). The matching probability of ConB is the same because of the same conditions. In order to generate the probability value of the new concept *car/bicycle* we sum up the probability value of ConA and ConB and divide it by the two because we are processing two probability values of the underlying concepts.

Definition 2.1. The probability value of one single concept is the number of matching relations in percent.

Definition 2.2. The probability value of the new concept in the upper Ontology is the sum of the probability values of the underlying concepts divided by the number of underlying concepts.

To ensure a high quality of the new concepts it is important to analyze similar Ontologies. Furthermore, the more Ontologies are analyzed the more important is the similarity of the Ontologies with the aim of receiving a high number of matches.

Another aspect which has to be considered is a functionality within the syntax of the upper Ontology. The used syntax has to provide the possibility to store probability values within the new concepts. Furthermore, when thinking about the given example of the new concept *car/bicycle* there should be the possibility to use this concept as *car* or *bicycle* in order to compare it with other concepts or terms. Therefore, the used syntax has to be extensible in a way that allows

the distinction within the name of the concept and it has to be able to store the probability value.

The compared concepts are stored in a new Ontology, the upper Ontology. For this we have to take into account that all concepts in the new upper Ontology are stored together with the probability value. With the probability value it becomes possible to expose the level of accordance when referring to the semantic knowledge structures of the new upper Ontology. At this point the importance of an adequate set of Ontologies becomes clear. If there are only few matches with a low probability value between the selected Ontologies the new upper Ontology will not be very beneficial. Therefore, the given set of Ontologies should consist of similar Ontologies which are related to the same research field.

For the purpose of ensuring the high quality of the set of Ontologies an Ontology Repository which contains the locations, respectively URLs, of the selected Ontologies will be deployed as well. The Ontology Repository was mentioned before but now the advantage of such a repository becomes explicitly clear. The Ontology Repository stores all required information regarding to the selected Ontologies in order to find them at a later iteration and with additional information about the type of Ontology. By this this the Ontology Repository stores more and more information about useful Ontologies and therefore a user of the Text-Content-Analysis is provided by this Ontology Repository when he is in the need of searching for adequate Ontologies. The idea of the Ontology Repository is to enable every user who has access to the Ontology Repository to add and change information. Hereby, all users of the aspired system will profit if the Ontology Repository is extended with new information or if it is updated.

2.2.2 TCA usage

The Text-Content-Analysis is the conclusion of the processes before. After a set of Ontologies is defined and the matching algorithm has compared the concepts of the given Ontologies the results of the Text-Content-Analysis can be used to provide with additional knowledge about an SLA.

With this analysis we intend to grant an understanding of several concepts by merging them to one concept. The new concepts can for example be used for the negotiation of Service Level Agreements. Therefore, Ontologies are selected whose scope is focused on mechanical engineering. Accordingly, the service oriented approach becomes clearer by the use of the Text-Content-Analysis. The analysis refers to the syntactic correlations of the concepts of the set of Ontologies as well as to the SLA-Management-System. Hereby, the developed software can be used as a extensible service. The aspirated method of Text-Content-Analysis enables a more detailed understanding of the content. In order to understand the content it is necessary to analyze the semantics as well. Hence, it is an important aspect of the procedure to consider the syntactic correlations as well as the content. But the main focus of the Text-Content-Analysis lies on the syntactic correlations. For

constructing this analysis formalisms for understanding textual content are needed. Through this formalism an understanding of textual content might become possible. Regarding to the introduced aspects of the assignment a textual understanding becomes essential. Based on the described procedure this textual understanding refers to the underlying Ontologies. The syntax within the Ontologies has to provide the ability of storing probability values. The Text-Content-analysis allows the interaction with the set of Ontologies, the SLA-Management-System in order to negotiate the requests of the users and the exposed formalisms. Through this, an issue of this work is focused on textual understanding. Therefore the Text-Content-Analysis becomes necessary. The understanding of textual contents from Ontologies whose scope is focused on mechanical engineering might be guaranteed through the Text-Content-Analysis method which will be deployed by the introduced procedure. Through the use of Ontologies that are constructed with RDF-(S) and OWL a wider range of Ontologies will be considered. With this the more open development of the described procedure leads to a wider range of applicability of the presented approach. Hereby, the use of standards such as XML, OWL and RDF-(S) which are common standards will support this essential aspect. Furthermore, the option to use the developed method in a wider range should be given. Therefore, the chosen standards have to support this issue.

When thinking about the interaction of the Service User with the whole system it has to be clarified in which way this should take place. For the Service User there are two connection points to interact. The one is the SLA-Management-System which gives the Service User the possibility to negotiate a SLA. The other point of access is the Text-Content-Analysis which provides the possibility for the Service User to select Ontologies which are comparable by the matching algorithm. Furthermore, the Service User is enabled to access and modify the Ontology Repository in order to find suitable Ontologies and to store new information (e.g. URLs) of Ontologies. However, the Text-Content-Analysis should be the first access point of the Service User in order to generate the required data for the semantic extension of the SLAs. With the Text-Content-Analysis it becomes possible to use the content of several Ontologies by using just the new upper Ontology. By that the possibility of managing several similar semantic knowledge structures within different Ontologies is provided. The Text-Content-Analysis method gives the opportunity to solve concepts which are distributed over several Ontologies by just using one Ontology. Certainly it has to be considered that the new upper Ontology is valid from its development until there are changes in the underlying Ontologies. Therefore, it is important to generate a new or to update the upper Ontology in periodic intervals. The simplest approach is to generate a new upper Ontology before every usage but when thinking of computing resources and processing time this will be the most expensive approach. Another idea is to check the upper Ontology during runtime in the background and informing the Service User about changes in the underlying Ontologies. However, if the Ontologies are checked permanently this procedure will be very expensive regarding to the computing

resources as well. Therefore, there are mainly two possibilities of handling these issues. The one is to accept the high amount of computing resources it will take to check the upper Ontology permanently or the other way is to accept having an upper Ontology which might be not up to date. However, if the upper Ontology is generated and it is used right after the provisioning then it seems to be acceptable to do no permanently update.

2.3 SLA-Management-System

The purpose of the SLA-Management-System is the negotiation of SLAs which are related to the specific needs of the Service User. The usage of SLAs and the negotiation supported by an SLA-Management-System is a well proven method for defining QoS aspects between two parties. Hereby, mainly economic requirements are negotiated automatically by such a system. As shown in the section before we have figured out and analyzed the relevant Ontologies. Furthermore, an Ontology Repository as well as a new upper Ontology is available now. Based on the upper Ontology which includes the matching results it becomes possible to make use of the new available semantic knowledge structures.

However, the aspirated SLA-Management-System refers to the upper Ontology and the negotiation of the SLAs is related to the concepts of the upper Ontology. Furthermore, the SLA-Management-System tracks a service oriented approach, which allows the user to interact with the SLA-Management-System in a extensible way by negotiation of the SLAs. There are mainly two different types of negotiation possible (Figure 5):

- Discrete Offer Protocol (Take-it-or-leave-it approach)
  This approach is about fix quality of service aspects. This protocol assumes that there is only one offer with static terms which is send to the user expecting his decision on acceptance.
- Multiphase Negotiation (n-phase)
  In contrast to the Discrete Offer Protocol the Multiphase Negotiation allows several rounds of negotiation. Hereby the n stands for the number of negotiation rounds. If n has the value one we are back at the Discrete Offer Protocol. In general there is the risk of blocking resources or running straight in an endless loop when not restricting the Multiphase Negotiation but this is a concern of the service provider. He has to clarify if he sets a limit to the negotiation phases or if there are other conditions of termination within the negotiation.

For the work presented in this chapter the used negotiation protocol is not a main objective because the focus of the work is the semantic analysis and the extension of SLAs by the extracted results. However, the choice of the negotiation protocol is a general decision which has to be made when thinking about the use of a SLA-Management-System and the given requirements.

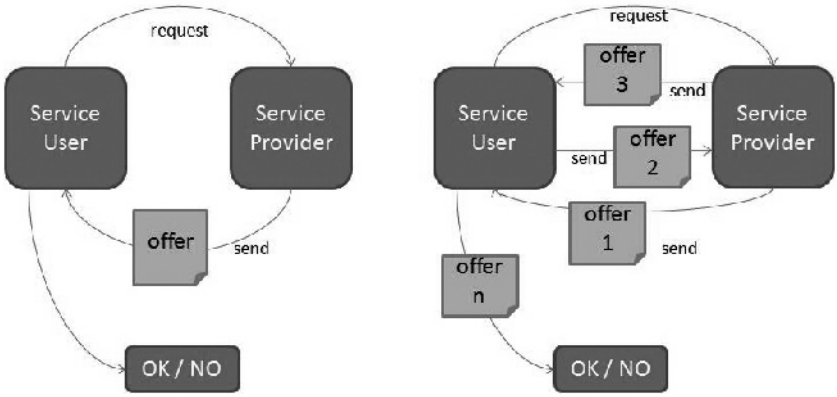

FIGURE 5. Discrete offer vs multiphase negotiation.

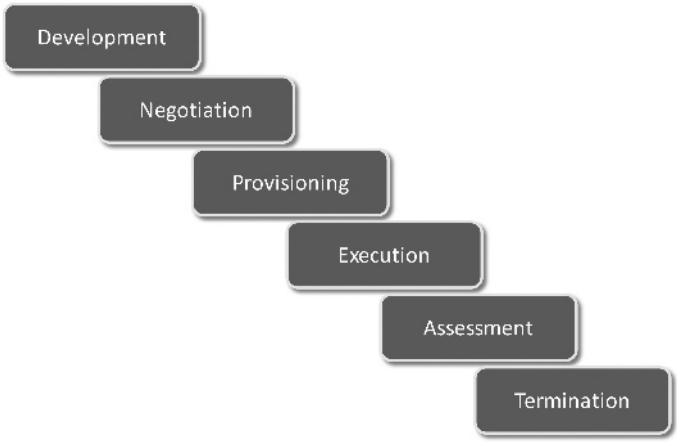

FIGURE 6. SLA lifecycle.

When we are talking about SLAs and offers we have to clarify the term of an SLA. First of all it is distinguished between several phases of an SLA, respectively the SLA Lifecycle. The SLA Lifecycle describes the development of an SLA until the termination (Figure 6).

- Development
  The first step of the SLA Lifecycle is the development of a service and the generation of one or more SLA Templates for the service. The SLA Templates are required for the publication of the features of the provided service. The generation of the SLA Templates is done by hand trough the Service Provider.

However, the Service Provider can choose to publish an SLA Template or not (in case that he has prepared special service templates for special Customers).

- Discovery & Negotiation
  The Discovery phase ensures that the Service User finds an adequate Service Provider. In general the Service User has access to a Service Registry which allows him to find a Service Provider by considering the requirements of the Service User. This approach is similar to the 'yellow pages approach'. The more precise the Service User describes his requirements the more useful are the results when searching Service Providers by the Service Registry. The requested requirements of the Service User are transmitted to the Service Provider. Therefore, the Service Provider is enabled to send an tailored offer to the Service Provider. If the negotiation is done and an offer is accepted an SLA is created.
- Provisioning
  The Provisioning of the service describes the supply and configuration of the system regarding to the SLA. When the negotiation of the SLA is finished the system has to be informed that there is a new SLA available.
- Execution
  The Execution phase takes place only when all steps before have been performed. The service is executed by using the negotiated SLA. Through the execution the current state of the service is monitored in order to intervene if there is a SLA violation.
- Assessment
  The Assessment phase takes place in parallel to the Execution phase of the Service. Through the Assessment the Service is monitored and evaluated during runtime. With this, violations can be detected and according measurements can be taken.
- Termination
  The end of the SLA Lifecycle is described by the Termination phase. If an SLA is not valid anymore it is terminated, the log-data are evaluated, the billing takes place and the used resources are reallocated. However, it is possible that an SLA is terminated during processing time if there are violations that cannot be fixed.

When thinking about the SLA Lifecycle and how to enrich the SLAs with semantic knowledge structures we have to consider that the semantic annotations of the SLAs are useful for the Service User during the Discovery and Negotiation phase. The semantic knowledge provides the Service User with additional information about the terms of an SLA in order to meet his specific requirements. Within an SLA-Management-System it is an economical need to ensure that the negotiated SLA fits with the requiremtns of both parties. Therefore, it is a need as well to ensure that the terms within the SLA are understandable by the user. This economical need is provided by the enhancement of SLAs with the extracted results from the described analysis method.

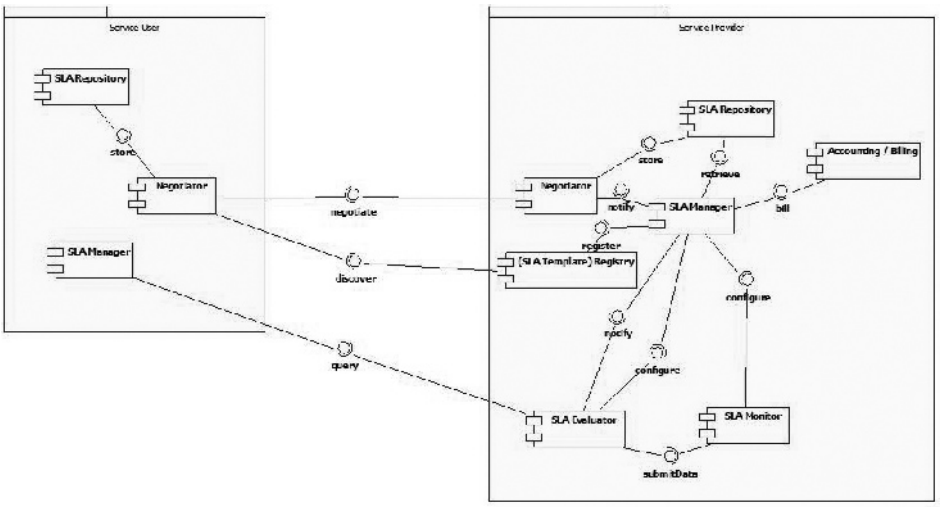

FIGURE 7. SLA-Management-System.

As mentioned before the SLA-Management-System should include a negotiation component, allowing for creating an agreement between Service User and Service Provider. Thereby the user requirements can be negotiated by providing semantic knowledge of the SLAs, which is additionally influenced by the Text-Content-Analysis maintaining this semantic knowledge. The SLA-Management-System is improved through additional information about the SLAs so that the negotiation can take place much more convenient for the Service User. By intention the Text-Content-Analysis is an extensible procedure which is governed by the individual user requests and his specific requirements.

It is obvious that the SLA-Management-System consist of more components than the SLA Negotiator. Further components are a SLA Repository, SLA Manager, Accounting/Billing component, SLA Template Registry, SLA Evaluator and a SLA Monitor. The meaning of the listed aspects is described in a brief way. The mentioned components are within the SLA-Management-System (Figure 7).

- SLA Negotiator:
  The SLA Negotiator is responsible for performing the negotiation between the Service user and the Service Provider. This component is existing in an instance on side of the Service User as well as for the Service Provider.
- SLA Repository:
  The SLA Repository contains the SLA Documents. Every SLA Document can be recovered from the SLA Repository by means of its unique identifier. Usually Service User and Provider have their own SLA Repositories.

- SLA Manager
  The SLA Manager of the Service Provider configures the SLA Monitor and the SLA Evaluator. The SLA Manager is able to handle decisions regarding a specific service by using incoming data. If there is a SLA violation the SLA Manager reacts, e.g. abort a service (worst case).
  On side of the Service User, the SLA Manager can as well react to events but his functionalities are less than the ones of the SLA Manager of the Service Provider.
- Accounting/Billing component:
  The Accounting/Billing component is responsible for the pricing of a specific service and for the billing when a service is done.
- SLA Template Registry:
  The SLA Template Registry manages stored SLA Templates. The SLA Negotiator is connected to the SLA Template Registry and thus he is enabled to receive already existing SLA Templates and to compare them to received requests for offers.
- SLA Monitor:
  The SLA Monitor reports values corresponding to specific metrics defined in the SLA Document. These values are required for the further evaluation of the service.
- SLA Evaluator:
  The SLA Evaluator analyzes the data received from the SLA Monitor. The evaluation report is send to the SLA Manager which is able to react based on the received data.

The listed aspects are an initial assemblage of what is required for an SLA-Management-System as for instance the FinGrid SLA-Management-System. Within the FinGrid project a SLA-Management-System for economical use in the financial sector is developed which shall allow for all necessary functionalities to cover the SLA lifecycle. This point is similar to the SLA-Management-System that is required for the Text-Content-Analysis. Through the interactive and extensible approach of the FinGrid SLA-Management-System it is possible to map the system to other use cases as well.

Based on the Text-Content-Analysis the negotiation of a SLA which is extended by semantic knowledge is ensured. Accordingly, there are three main topics that need to be developed in a connectional manner:

- Analyzing Ontologies:
  evaluating the Ontologies; analysis of the syntactic correlations between Ontologies and concepts; development an upper Ontology based on the matching algorithm;
- Text-Content-Analysis:
  implementation of the Text-Content-Analysis;

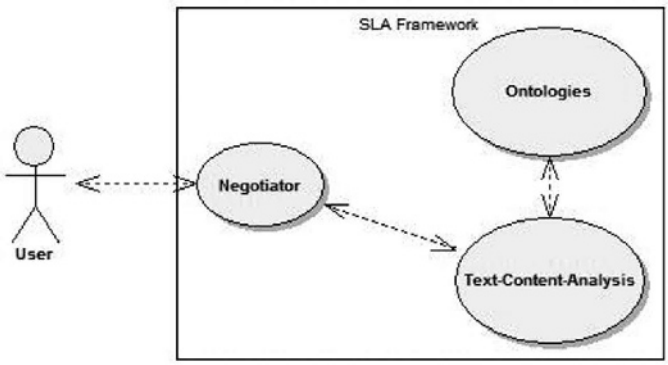

FIGURE 8. Usage of the system.

- SLA-Management-System:
  deployment of the SLA-Management-System; negotiation of the requirements of the Service Users and the concerns of the Service Provider;

In order to merge the three main tasks of this work the next section will give a brief overview about the usage of the maintained service.

2.4 A useful service

The described process enables the interaction with the implemented system as a service. Through the SLA-Management-System the Service User is able to use the developed method of Text-Content-Analysis in a way that is specified to his own requirements. The Users request is negotiated through the deployed SLA-Management-System (Figure 8).

The negotiation component which is within the SLA-Management-System refers to the Text-Content-Analyze method. However, the method refers implicit to the set of Ontologies through the upper Ontology which includes the semantic knowledge structures of the underlying set of Ontologies. The whole service for SLA negotiation is embedded in the SLA-Management-System and the Ontologies as well as the Text-Content-Analysis are connected to the SLA-Management-System. One basic issue of the project is to enable a Service User to negotiate SLAs by using semantic knowledge. Therefore, the project includes not only constructing the Text-Content-Analysis based on the syntactic correlations between Ontologies. It includes a service concept as well. The development of the SLA-Management-System including the set of Ontologies, the matching algorithm and the Text-Content-Analysis let the described work within this chapter become a service oriented system that might be used in a way that enables user requests to the service via internet. Hereby, many users are able to use the service through SLA Management Service and the Text-Content-Analysis.

3. Conclusions

Through the development of the described procedure a service with a higher extensibility for the usage by the Service User is achieved. At this the negotiation component is a significant part for the service. The quality of the service is ensured by the negotiation so that the requirements of the Service User are considered. Therefore, the Service User support through the new available semantic knowledge structures is a main task of the aspired work in order to guarantee the high standards of the negotiation. Through this a Service User who does not have to be an expert in the current research field is enabled to negotiate a SLA. The additional semantic knowledge supports the Servce User in very convenient way. By that we also see an increase of the economic use of SLAs as Service Users with the extracted knowledge structures are well supported.

Moreover, the realization of the presented work enables a textual understanding of content based on the syntactic correlations between the concepts of a given set of Ontologies. For this approach a syntax and therefore formalisms have to be developed. The syntax has to support the storage of probability values. The development of such formalisms and syntax might be of great interest to the field of language understanding. However, the described work might be interesting to the field of information system analysis and economic studies of SLAs as well. The Text-Content-Analysis which refers to the Ontologies permits an understanding of the textual contents by analyzing the syntactic correlations. Hereby, a language understanding based on Ontologies is modeled. The introduced work of this chapter permits an interaction between Service User and system by negotiation of the users' requests. Hereby, the system is used as a service. The development of the service via internet contributes to the future of the internet and makes the system available for a wide range of users.

Acknowledgements

This work has been supported by the BREIN project (http://www.gridsforbusiness.eu) and has been partly funded by the European Commission's IST activity of the 6th Framework Program under contract number 034556. This work expresses the opinions of the authors and not necessarily those of the European Commission. The European Commission is not liable for any use that may be made of the information contained in this work.

This research is a part of the Financial Business Grid project (http://www.fingrid.de/), coordinated by the E-Finance Lab at the J. W. Goethe University, Frankfurt, Germany. This material is based upon work supported by the German Federal Ministry of Education and Research under Grant No. 01IG07004D. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the D-Grid Initiative or the Federal Ministry of Education and Research.

This work has been supported by the LarKC project (http://www.larkc.eu/) and has been partly funded by the European Commission's IST activity of the 7th Framework Program under contract number 215535. This work expresses the opinions of the authors and not necessarily those of the European Commission. The European Commission is not liable for any use that may be made of the information contained in this work.

References

[1] The BREIN Project, Website http://www.eu-brein.com/.

[2] The FinGrid Project, Website http://www.fingrid.de/.

[3] The LarKC Project, Website http://www.larkc.eu/.

[4] GRAAP-WG, OGF, http://forge.ogf.org/sf/projects/graap-wg.

[5] WS-Agreement: https://forge.gridforum.org/sf/go/doc6091?nav=1.

[6] Web Service Level Agreements (WSLA): http://www.research.ibm.com/wsla/.

[7] B.-Y. Ricardo, R.-N. Berthier, *Modern Information Retrieval.* ACM Press Books. New York, 1999.

[8] D. Oberle, *Semantic Management of Middleware.* Springer Science+Business Media, Inc. NewYork, 2006.

[9] T. Pellegrini, A. Blumauer, *Semantic Web.* Springer-Verlag. Berlin; Heidelberg [u.a.], 2006.

Axel Tenschert
The University of Stuttgart
High Performance Computing Center Stuttgart
Nobelstraße 19
70569 Stuttgart
Germany
e-mail: tenschert@hlrs.de

Ioannis Kotsiopoulos
The University of Manchester
Kilburn Building
Oxford Road
Manchester, M13 9PL
United Kingdom
e-mail: ioannis@cs.man.ac.uk

Bastian Koller
The University of Stuttgart
High Performance Computing Center Stuttgart
Nobelstraße 19
70569 Stuttgart
Germany
e-mail: koller@hlrs.de

Part III:
Business Models and Market Mechanisms

Economic Models and Algorithms for Distributed Systems, 183–185
Book Series: Autonomic Systems

Business Models and Market Mechanisms

Business Models and Market Mechanisms Business IT management has seen limited use of Grid computing concepts in its day-to-day operations. This is largely due to the voluntary, open-source nature of Grid computing. Similar to the open-source movement, Grid concepts do not conform to most profit-oriented approaches. It is often due to reasons such as determining the cost of one hour of computing for instance, which makes Grid computing so hard to couple with business approaches. Nevertheless the technological and conceptual feasibility of using Grid computing is evident, as it has worked relatively well in research environments.

Cooperative computing infrastructures do not only change the way traditional software systems are used, but also facilitate the establishment of new business models. While licensing models of software diminish in importance, on-demand services become more and more attractive. With the advent of cooperative computing, there is a shift from technologies towards intelligent business models. One reason why Grid computing has failed so far, despite the viable promises of the technologies, is the lack of effective business models. The voluntary contribution model may hold for e-Science communities, however it is not so applicable in a commercial context. The movement from Grids to Clouds addresses these weaknesses by offering Cloud services by one single vendor. In these cases, the liability question is much more obvious in Clouds than in the integration of Grids. The concept of Clouds is currently taking over the business orientation from utility computing and expressing the shift of IT from being an asset those companies possess (e.g. computers, and software) to being services that companies purchase from designated Cloud providers. Utility computing introduced the service provisioning model in which a service provider makes computing resources and infrastructure management available to the customer as needed, and charges them for specific use.

Service provisioning and charging depends on the processes defined in a business model. The emphasis of utility computing rested on two general pricing models: the first pricing model is the subscription based model. Accordingly, users are charged on a periodic basis when they subscribe to a service. The second model is the metered model, where users are charged on the basis of actual use of resources. The choice of the pricing model clearly has an impact on the

incentive structure: while the first pricing model gives rise to a waste of resources, since charging is independent of usage, the metered model is ultimately deemed the promising alternative that removes the waste of resources (Rappa 2004). From the users point of view, the metering of usage can be compelling, as one pays for what one uses. For the service provider the metered model is appealing, as it offers an opportunity to present idle or unused computer capacity. The metered model, however, is not a panacea. It has to be determined, by what price a company charges and when the meter is turned on (Rappa 2004). The price is essential, as it determines the incentive for resource owners to provide them as well as for consumers to demand use of resources. If the price is too high, not all the offered resources are demanded, which contributes to idle resources. If the converse is true, not all tasks can be realized as the resource provision is too low, which is equally bad.

A lot of attention has been devoted to market-based approaches in cooperative computing environments, as these settings are assumed to work well for price determination. By assigning a value to their service requests, users can reveal their relative urgency or costs (Buyya et al. 2004; Irwin et al. 2004). Despite their theoretically useful properties, only few market-based approaches have been implemented in operational systems, let alone commercial ones. In this section, business models and in particular market mechanisms are explored, focusing on how such mechanisms can be designed to support cooperative computing environments. In addition to the charging issues, which dominate utility computing, business models of cooperative computing environments are becoming very versatile in how the services are being provisioned.

The first article *Cloud Computing Value Chains: Understanding Businesses and Value Creation in the Cloud* by Ashraf Bany Mohammed, Jörn Altmann and Junseok Hwang explores potential business models for cooperative computing environments in a broad perspective. In their attempt to promote cooperative computing environments under the umbrella of Cloud computing, the authors home in on problems associated with creating Cloud value chains, similar to standard business supply chains. The authors underline the possible service activities, based on a case study that contributes to assigning value to computing services. Furthermore, they give some insight into the flow of information and payments within the value chain structure, extending insight to some important factors such as cost-minimisation, profit maximization and return on investment, for both resource providers and consumers.

Following this broader value approach, the second article by In Lee focuses on the specific question of optimal capacity investment in utility computing as a fundamental part of business models for cooperative computing. While the previous article focused on valuations within the Grid itself, this work presents the view from the outside, focusing on the resource providers, and their capacity decisions in *A simple Model for Determining the Optimal Capacity Investment for Utility Computing*. It presents a unique insight into utility computing and

capacity planning, focusing on states of under- and overcapacity, as well as the decisions faced by providing an economic model. With utility computing evolving from a cost-oriented approach to a strategy-oriented demand approach, economic utilisation of resources has become a top priority. This article shows that resource over- or under-utilisation, although posing problems for the enterprise involved, also provides opportunistic situations for cooperative computing.

A Combinatorial Exchange for Complex services focuses on a market-based approach for trading Grid services. Contrary to former work, where value was determined by looking at the value chain, this work allows the determination of the value by market-based demand and supply mechanisms. This approach is designed to maximise the social welfare of all parties involved in trading on the Grid market. Unique to this model is its incentive compatible design, giving resource owners a realistic incentive to offer their idle resources via the Grid, using prices to aggregate and disseminate information about the system's status and help to control the dynamic demand and supply. This work also focuses on scenarios in which resources are scarce, presenting a scheduling mechanism which still holds true for the goal of social welfare maximisation.

Even advanced optimisation concepts for the operation of distributed infrastructures, incorporating social welfare considerations, generally do not include ecologic externalities. However, this is particularly relevant in the age where global warming has become a great threat to our society. In regular markets these aspects have been included with the help of e.g. Pigou-taxes aiming to regulate CO2 emissions. Computing, as an end-user of electricity, cannot be held accountable for the emissions in power production. *Heuristic Scheduling in Grid Environments: Reducing the operational energy demand* focuses on green issues by incorporating energy efficiency, without losing too many economic incentives in the optimization of scheduling plans in an offline environment. By internalising energy consumption into the assignment problem, an economic optimal as well as energy efficient schedule can be derived. For an offline scenario, with full information, the allocation mechanism derives optimal schedules. Nevertheless for online scenarios, with incomplete information and advanced reservation, ordinary pricing strategies are no longer sufficent.

Selling Grid services is not always done in a timely manner. Resources are often requested in advance, making the development of incentive compatible pricing strategies a non-trivial problem. In *Facing Price Risks in Internet-of-Services Markets*, uncertainty is modelled as a price risk, applying methods from financial markets. This work aims firstly at identifying the difference between financial and Cloud computing markets and their risks before applying and adapting option pricing models for computing environments.

Economic Models and Algorithms for Distributed Systems, 187–208
Book Series: Autonomic Systems

Cloud Computing Value Chains: Understanding Businesses and Value Creation in the Cloud

Ashraf Bany Mohammed, Jörn Altmann and Junseok Hwang

Abstract. Based on the promising developments in Cloud Computing technologies in recent years, commercial computing resource services (e.g. Amazon EC2) or software-as-a-service offerings (e.g. Salesforce.com) came into existence. However, the relatively weak business exploitation, participation, and adoption of other Cloud Computing services remain the main challenges. The vague value structures seem to be hindering business adoption and the creation of sustainable business models around its technology. Using an extensive analyze of existing Cloud business models, Cloud services, stakeholder relations, market configurations and value structures, this Chapter develops a reference model for value chains in the Cloud. Although this model is theoretically based on porter's value chain theory, the proposed Cloud value chain model is upgraded to fit the diversity of business service scenarios in the Cloud computing markets. Using this model, different service scenarios are explained. Our findings suggest new services, business opportunities, and policy practices for realizing more adoption and value creation paths in the Cloud.

Mathematics Subject Classification (2000). Primary 91A40, 68M14, 68T99; Secondary 91A10.

Keywords. Ontologies, semantic concepts, semantic knowledge structures, semantic enhancement, ontology matching, service level agreements, negotiation, SLA-Management-System

This work was completed with the support of ITPP center – MOKE – Korea

1. Introduction

The IT market is evolving quickly, driven by the increasing need for costs cuts and more agile and effective business processes. Cloud computing emerged as a promising computing model for providing utility-based, on-demand IT infrastructure services for anyone, anywhere and anytime [1, 2].

The developments realized in the past few years in computing techniques, especially in Grid computing, enabled the emergence of numerous computing models: Utility computing, ubiquitous computing, cyber-infrastructure, e-science, e-infrastructure and, above all, Cloud computing. Although many believe that these Cloud-based technologies hold the potential to revolutionize the Internet [3], actual adoption of Cloud computing services in industry and business is still way under expectations. It seems that the transition from classical enterprise IT models to Cloud-based computing is still the biggest challenge in businesses and industry, despite all the advancements that supported this transition.

Complexity in existing value structures and modes for attaining cost efficiency practices are main shortcomings in the Cloud. These problems are believed to be the chief factors in contributing to the weak business and industry deployment, low adoption, and missing sustainability.

Understanding the structure of the Cloud and its potential value creation schemes is challenging due to the diversity in requirements, inherited technical complexity, and unstructured service schemes. This complexity made the provisioning and utilization of many Cloud technologies and services a very difficult task to anticipate, especially by non-IT businesses. For example, the deployment of services, composition of services, and troubleshooting of many Cloud services needs special preparations even for IT experts. In addition to this, top management officers in enterprises or governments need to understand how costs are accumulated and how value is added without going into deep technical details of deployment or provisioning. Therefore, clarifying the value structure and corresponding primary and support activities in the Cloud value chains would help both, the business and the Cloud community, in accelerating adoption and creating more value.

Following the line of argument, this paper aims at addressing the following questions:

- What is the value structure in the Cloud market? What are the "primary and support" service? What activities that contribute to value creation in the Cloud?
- How do money and knowledge flow between stakeholders? And how can we minimize cost, maximize profit, and increase return on investment for Cloud stakeholders (providers and consumers)?

In order to tackle these questions, this research uses a case study analysis. Cases were reviewed and analyzed from available Grid and Cloud market services, projects, tools, applications, business models, and technologies. Based on the

value chain theory, this analysis is used to create a reference model for Cloud value chains.

From the provider's perspective, the proposed Cloud value chain model helps Cloud Service Providers (CSP) to realize where they stand in the Cloud market and how they relate to other CSP. It particular, it helps CSP to identify their needs, anticipate potential alliances and create new service provisioning scenarios. This would also facilitate new entrants understanding of potential markets, formulate their value model based on market needs and fully utilize existing services. From the consumer's perspective, consumers will be able to identify the different potential costs for using and customizing the Cloud based on their business needs, and foresee diverse service scenarios from a strategic point of view.

To ensure the integrity of the model, the Cloud value chain reference model is later validated against hypothetical business scenarios. The validation also considers the value creation process and the flows of money and knowledge between the different stakeholders in the Cloud. As a result, a clear differentiation between main and support activities has been achieved, highlighting potential costs and opportunities in the Cloud.

The remainder of this chapter is organized as follows: In the next section, we review the literature on value chains and present the state-of-the-art in Cloud value chains. Section 4 describes the Cloud value chain reference model while Section 5 provides the business scenarios. Section 6 draws policy recommendations and Section 7 concludes this work.

2. Literature review

2.1 Porter value chain

The value chain was first described by Michael Porter in [4]. According to Porter, the value chain is a "system of independent activities, which are connected by linkages. Linkages exist if the way, in which one activity is performed, affects the cost or effectiveness of other activities". Linkages illustrate how a single activity affects other activities, thus serving as an important source of competitive advantage and value adding [5]. The value chain classifies activities into primary activities and support activities. Products (i.e. services) should pass all activities of the chain sequentially and, at each activity, the service gains some value and the chain of activities gives the product more added value than the sum of all single activities [6].

2.2 Upgrading the value chain

In the Internet era, value chains bear more advanced networking capabilities, adaptability to external changes, accessibility to virtual company structures, and more dynamic management capabilities [7]. In fact, many industries' value creation processes are dynamic, flexible, non-linear and multi-dimensional, crossing

horizontal and vertical (inter- and intra-) industry domains. Porter's original value chain is believed to be linear and fixed [8–10]. Consequently, Value Networks [8] emerged to provide a platform for modeling non-linear complex set of social and technical resources, working together via relationships.

In those value networks, value is created through the exchange of resources via relationships between roles. Although value networks represent a new approach to describe tangible and intangible value creation processes [6, 8], value networks modeling becomes complex, since the structure grows exponentially. This leads in many cases to many difficulties in foreseeing economic competitiveness, new opportunities, or anticipating costs.

A more advanced concept, known as "value grid" (the grid is not to be confused with Grid Computing), has been introduced in [9]. This model allows expressing innovation strategies and coordinated operations in multidimensional, grid-like value creation schemes. In fact, the "value grid" extends the concept of value chains. In a value grid, the vertical dimension describes multiple tiers from primary inputs (raw materials) to end users. The horizontal dimension describes opportunities at the same tier across parallel value chains; and the diagonal dimension describes opportunities for integrating value chains [9]. This gives the value creation model an evolving structure, where the flow of knowledge and intangible benefits can take place between any number of stakeholders anywhere and anytime. At the same time, it maintains an organized and systematic structure.

In general, whatever representation for value creation is being used, value chain models should allow all stakeholders to explore the diverse scenarios for generating value and to find their unique added value, guaranteeing their competitiveness, the maximum return on their investment, and their economic sustainability.

2.3 State-of-the-art: Cloud value chains

A number of projects studied different aspects of value chains in the "information technology industry". Most recently, the "BEinGRID" project developed a Generic Grid Computing Value Chain and a corresponding value network model based on the analysis of industry case studies [10, 11]. In their classical Generic Grid Value Chain, different Grid stakeholders are described.

However, the organizational aspect of value creation has been overlooked and more attention has been paid to value networks. Same overall value creation process in the Grid technology market was also reported in [12]. Although it addressed the complexity of the value creation in the Grid industry in its early days, this model can not capture the Cloud market and suffered from same drawbacks as described in [10].

Another related value chain model has been created and analyzed for the ubiquitous computing environment [13]. Yet, within this work, the abstraction of the value chain model to five stages (context, information devices, networks, service providers, and digital content) is too simplistic. Despite this pioneering work, the linearity and highly abstracted representations were not able to proxy the value

creation processes and potential value creation paths in the Cloud. Finally, a related work presented a detailed taxonomy and description of the stakeholders and their roles in the Grid [14]. Yet, this work was not oriented towards value chains and Cloud computing.

In summary, although these models provide a solid foundation for the development of Cloud value chains, they were not cloud-centric and none of them depicts the value transactions in an organized and systematic scheme that accounts for all stakeholders' inter-relations and value creation structures in the Cloud.

3. The cloud value chain reference model

3.1 Data and methodology

The data for the value chain analysis was collected through market surveys and case studies available online at Cloud, Grid and Internet service providers, project Web sites, descriptions of tools for Cloud computing, application specifications, and business models of service providers. A data set comprises information about the product and/or service, price, business linkages between partners and suppliers, cross layer linkages, as well as a description of the value creation process. Using value chain analysis techniques of Porter [4] and Hergert & Morris [31], this data was processed as follows:

- Set the boundaries of the business segments.
- Defined the critical activities accordingly.
- Identified the product (services) value structure.
- Positioned the linkages and inter-relationships within the relative structure.
- Set the activities within the context of original Porter value chain model following upgraded value chain guidelines.

Based on this procedure, the linkage, the layers, and sub-layers were set for each service. A fraction of this list of services and products is given in annex 1, the complete is list is available on our Web site.

3.2 Model

Building on the foundations of Porter classical model, value networks, and "value grids", the following Cloud value chain reference model was developed. The model breaks activities (services) down into three main virtual layers. Within those layers, services are organized as independent sub-layers. Each layer border represents a profit and knowledge margin. Linkages between layers or independent services can take horizontal, vertical, and diagonal paths. Value is accumulated by flow of money and knowledge through these linkages. Through this organization, service packages can be created in a flexible and cost effective way. Figure 1 shows our proposed Cloud value chain reference model.

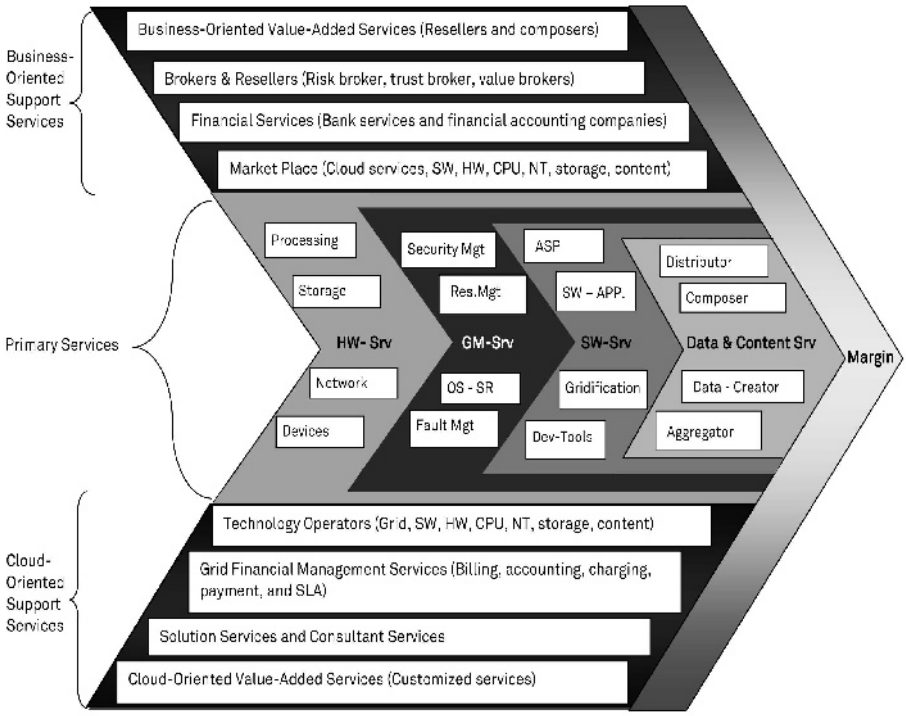

FIGURE 1. Cloud value chain reference model.

Primary (Core) Services Layer This layer incorporates core infrastructure services, required for the development of any Cloud service model (enterprise Grid, high-performance computing Grid, business Cloud, or public computing Cloud). The structure of this layer resembles the Open Grid Services Architecture (OGSA) [32].

This layer consists of the following sub-layers: The hardware services (HW-Srv) or fabric sub-layer includes networking, processing, storage, and other device services. Generally, a wide range of these services is provided as standalone systems from classical technology providers such as Sun Microsystems, IBM, CISCO, and HP.

The Grid middleware services (GM-Srv) layer is commonly considered the founding Cloud layer. This layer mainly includes resource management services (R.Mgt.-SR), security, privacy, fault management, Grid execution, and operating systems services (OS-SR). Services in this layer range from complete packages such as Globus [15, 16], Legion [17], gLite [18, 19], UNICORE [20], Xen [41], and Grid Application Toolkit (GAT) [34] to standalone system services.

As the fabric sub-layer services, the software sub-layer services are usually provided by countless software companies, ranging from big players to an open source software community. The software services (SW-Srv) sub-layer includes software applications (SW-APP), Gridification services (e.g. Grid Execution Management for Legacy Code Architecture – GEMLCA) [33], developing tools (Dev-Tools) and application support services offered by Application Service Providers (ASP).

Finally, the data and content services (Data & Content-Srv) sub-layer includes data creation, aggregation, and distribution services. Data libraries, 3D & multimedia data, and research databases are some of the technologies in this layer. Providers for such content services can vary from professional data service companies, universities, and research centers to end-users.

Note that all of the primary layer and sub-layer services are provisioned as packages or independent services, outsourced and/or built in-house. Deciding for any of these approaches always depends on the anticipated opportunity cost and the value added to the provider or customer.

Cloud-Oriented Support Services Layer This layer incorporates all activities developed solely to support and enable the Cloud in real world markets. Services in this category are normally developed and deployed based on customized or specific needs. Whereas some Clouds need all these services others may need just few. In many cases, services from this layer are bundled with core activities (particularly in Grid middleware sub-layers) and are provided as a full package offered by Cloud Service Providers.

The support services layer consists of the following service sub-layers: a) Cloud financial management services. They incorporate basic Cloud-based charging, accounting, billing, payment, and SLA management systems. An example of these services is SLA@SOI (see appendix); b) Solution providers, consultant, and composer services.

Examples of this type of service include infrastructure for service delivery from RESERVOIR (see appendix); c) Technology operator services. These operators do not own the resources but operate the resources for the benefit of another party. Their services include activities of running, operating, and troubleshooting core services (e.g. software, hardware, network, storage, and content services). This service is especially important for big businesses and industries; d) Cloud-oriented, value-added services (customized services). This layer represents support services that address emerging needs in the market by supplying niche services. An example of this service could be customization of existing support services.

Business-Oriented Support Services Layer This layer represents all (non-tech nical) business services that support businesses and Cloud industry business. This layer includes services in existing, real-world markets, customized to the business side of Grid computing.

Some of these business support services (e.g. banks and brokers) exists since a long time and served many industries. These activities include the

following sub-layers: a) Financial management services (e.g. banks and financial solution companies); b) Brokers and resellers services (e.g. sellers, distributors, insurance companies); c) Marketplace services such as GridEcon [21, 37, 40], Gridipedia [11], and Amazon [22]; and d) Business-oriented value-added services, which are anticipated for innovative future business services. These include a wide range of systems. A good example of such services is TXTDemand, a demand forecasting system provided by SORMA (see appendix).
A detailed description of many of these services and stakeholder roles can be found in [14, 38].

3.3 Model structure and relations

To maintain efficient processes, structures, and linkages in value creation and service scenarios, the reference model holds the following properties:

- Structure and Organization: The model is made up of a virtual layer of services and sub-layers, organized according to core, value-added, and support services. This will help maintaining readability and structure in value creation process.
- Inter-Layer Heterogeneity and Intra-Layer Homogeneity: Layers and sub-layers make services share common knowledge bases, comprise strong relationships, carry out related functionality, and serve closely related goals. Consequently, spillovers, bundling, and packaging has more probability to take place between direct and nearby layers and sub-layers.
- Knowledge Flow: Services of each layer and sub-layer have knowledge spillover effects that span beyond its borders, allowing value to be shared and exchanged between them. Knowledge flows will add more value and help minimize the costs. For instance, the flow of knowledge between solution providers and financial service providers or between brokers and market place service providers can have these effects.
- Flexibility in Service Composition and Value Creation: Each Grid value creation process or service scenario has its own distinctive and different requirements. Therefore, the model relaxes the linearity condition of the Porter model and allows vertical, horizontal, and diagonal combination of services. This enables anticipation of service scenarios that are simply based on value and cost effectiveness.

Figure 2 shows an instance of the proposed Cloud value chain reference model. In particular, it shows sample services available in the market in each layer of the value chain. Each service is associated with a price and attributes to meter the total cost. Service costs vary according to the service type, quality, and usage model. By matching the services available in the market to the value chain, customers or providers will be able to foresee different scenarios for deploying or developing their Cloud services.

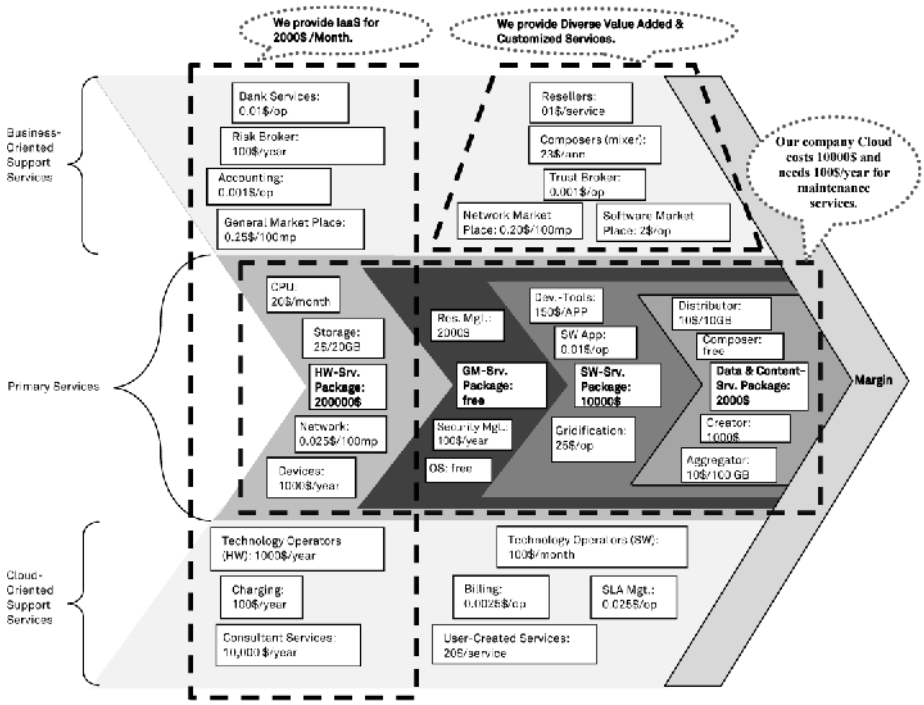

FIGURE 2. Instances of cloud value chains.

In detail, Figure 2 illustrates three composed services that are composed of sample services. For instance, the vertical, dash-lined rectangle shows an example of a service, which is composed of a set of infrastructure services and which costs 2000$/year. This service consists of: hardware (storage and devices), maintenance and operation, charging, and consulting, belonging to the primary and Cloud support layers. These costs are added to the costs of brokers, accountants and market services. Another service is illustrated in the upper right corner of the figure, which offers a wide set of customized and user-oriented services. Finally, the horizontal dash-lined rectangle illustrates an enterprise Cloud service that integrates all primary services. As these examples show, examining services with the proposed Cloud value chain reference model will help stakeholders in understanding value structures and foresee opportunities, while minimizing costs.

3.4 Service scenarios

To illustrate the workings of the Cloud value chain reference model, this section describes five different Cloud value creation scenarios that are explained based on the reference model. These scenarios are either existing Grid service scenarios or are potential future scenarios.

3.4.1 Utility cloud

In this scenario, services are provisioned in a similar way to any other utilities (electricity, gas, and telecommunication). The Cloud service provider offers a complete end-to-end package of Grid services that consist of primary, Cloud support, and business support services to fulfill the requirements of a targeted class of users. Services here are provided on-demand and metered based on consumption parameters (quantity and quality). Therefore, the user role here is limited to plug-and-play with no additional huge fixed investments or worries about complex structure, maintenance, or support.

Validating this scenario against the Grid value chain model, we find that the Utility Cloud Service Provider (UCSP) needs to carry out all primary activities, which include hardware infrastructure (bandwidth, servers or other special devices needs), Grid middleware (resource management system and security), and software services. Besides, the UCSP needs to provide financial services for the users of his Cloud service (such as charging, Service Level Agreement (SLA), and metering schemes). At the same time, the UCSP has to provide business support activities and define payment and billing services, broker's services, and market services. The first generation of such a UCSP (since much more comprehensive services can be offered in the future) is Amazon's Elastic Compute Cloud (Amazon EC2) and Sun's Grid Compute Utility Service [22, 23].

3.4.2 Enterprise cloud

Large enterprises (e.g. "Wal-Mart, which have more than 400-billion-row tables, which ultimately top a trillion rows" [24]) require huge computing power for intensive data analysis. Enterprise Clouds offer a cost effective solution for utilizing the internal computing resources to satisfy this huge demand. In this scenario, the Cloud is constructed for supporting internal business needs for the enterprise. The enterprise in this scenario has its own hardware, software, and business applications. That means that much of the primary activities are already in place. With dependable Grid middleware systems, which can be either in-house developed or outsourced from technology providers along with Gridification systems, the enterprise Cloud will be basically operational. Nevertheless, some business and Cloud support activities (e.g. consulting services, wrapping services, and SLA services) will still be necessary to purchase from the Cloud.

The final decisions will depend on enterprise value models and vary from case to case. Nevertheless, using Cloud value chain modeling will help technology officer to foresee their costs and justify their arguments to executive management. During the past years, many firms transformed their information systems to Clouds. Charles Schwab, the financial services provider has been using IBM servers with Globus middleware for its Grid for a while. Acxiom, a global marketing services firm, is using Red Hat Lunix operating system and JBoss Enterprise Middleware for its Grid processing environment. A list of other examples can be found in [25].

3.4.3 Research grids

In this service scenario, the Grid is constructed to support academic and research communities. Validating this scenario against the value chain model easily shows that most primary activities such as hardware resources, networks, and many other requirements are already in place. However, middleware is needed, which is developed internally in most cases rather than outsourced to the market (not considering that the research community collaborates and shares their middleware). Since applications are developed in-house as well, no additional cost is incurred. Access and usage policies are usually set on non-monetary bases (mutual sharing of resources). Consequently, services and systems are required from both, the business-oriented support layer and the Cloud-oriented support layer. For instance, while financial and broker services are not required, a clearing-house service is needed to balance accounts with respect to contributions and consumptions. However, in variation of this scenario, if research Clouds become open for businesses in the future, financial accounting, metering, and charging services will be needed.

Many examples of research Grids exists. Some of the notable examples include: TeraGrid, EGEE, LA Grid, and D-Grid. The TeraGrid is a scientific research infrastructures launched by the National Science Foundation in the United States. TeraGrid provides more than 250 teraflops supercomputing services capability, 30 petabytes data resources, as well as high-end experimental facilities for researchers from diverse disciplines. The Enabling Grids for E-science (EGEE) is an EU project connecting more than 240 institutions in 45 countries world-wide. EGEE infrastructure provide seamless computing services for e-Science that consists of 41,000 CPU in addition to about 5 PB of storage, and maintains 100,000 concurrent jobs [18]. The LA Grid is a joint Grid effort of the United States, Latin America, and Spain [26]. At a national level, the German Grid Initiative (D-Grid), which has been launched in 2004, is comprised of a number of projects that designs, builds, and operates a network of virtualized high-performance resource services [27].

3.4.4 Public clouds (desktop grids)

In this scenario, end-users collaborate in constructing their computing resources. A user-collaborative Cloud is built of what is referred to as the collective power of members. The primary activities consists of user PCs (being the main computing resources), the Internet (being the backbone network), and the freely downloadable middleware (BONIC) and application. Based on intangible incentives for participants with very little Grid supported services from project moderators (who are researcher and volunteers) and with no business-based services except SLAs, this model showed a huge success. SETI@home, a public desktop Cloud, has been standing as a symbol for successful sharing of distributed computing resources in

search for extraterrestrial intelligence [28]. Following this success numerous similar projects were initiated worldwide (e.g. Einstein@home [29], Rosetta@home [30]).

3.4.5 Virtual clouds (VC)

This scenario describes a future service scenario, where the Virtual Cloud Service Provider (VCSP) is not the owner of any physical resources. The VCSP builds his value-added service by composing services from different providers and create his "Cloud services". Similar to the business model of Application Service Providers (ASP), services are provisioned on-demand to customers. Yet, in this scenario, services have a wider scope (hardware, software, applications, content). In this case, the knowledge of organizations and the management of services is crucial. Issues such as quality of service, security, and privacy would be top concerns. Yet, ideally, the user should experience a plug-and-play service without any worries about the underlying details, which will be the responsibility of the VCSP.

The Cloud value chain reference model gives a comprehensive picture of the whole market structure and supports VCSP in understanding and identifying the diverse requirements at each layer, enabling the VCSP to build a working Cloud service. Moreover, the VCSP can anticipate his potential costs, foresee how his revenue model is structured, and improve his business model by optimizing alliances with other service providers.

4. Discussion and policy implications

The reference model for value chains in the Cloud is anticipated to serve as a check-list for building Cloud services. By validating this model against existing and hypothetical service scenarios, this model could largely explain the different types of activities and service combinations by which value is created in the Cloud. Yet, this model assumes and requires that:

- More and more activities in the Cloud are developed and introduced as independent, interoperable, and standardized services, allowing to realize flexible, diverse, and cost effective service scenarios. In fact, the work of Spohrer in [35] provides a promising effort that explores this mutual understanding, the value co-evolution, and the merging of service science with new computing models.
- While vast attention has still to be paid to technical layer development, more and more business-support services are developed [39]. For example, Cloud market places need to be in place to support trading of excess resources [36]. Accounting and charging services are needed to support financial settlements between trading entities (i.e. providers and customers). In general, business-support services are needed to accelerate the transition to a business-oriented, open Cloud to generate the critical mass of users.

Realizing such a future Cloud, we can forecast the following implications for the future technology market with respect to:

- Supporting Services: In accordance with the "hourglass" model [16], we strongly believe that support activities will take the highest share of cost and profit, dominating over middleware system services.
- Service Composition: Better cost minimization and profit maximization schemes will come from services that allow diverse compositions and service scenarios built from sub-layer services.
- State-Less Stakeholders: Customers will easily be able to become service providers. Users will be able to cooperate with other users and sell user-created services into the market. Future killer services will be user-created.

5. Conclusion

Within this chapter, we analyzed existing Cloud business models, its structures, and value creation activities. Based on this analysis and the existing value chain theory, we introduced a reference model for the Cloud value chains. The reference model is used to anticipate the Grid structure costs and service scenarios.

With the help of the reference model, we analyzed some service scenarios. We found that, as more standardized, interoperable and open technology services are realized, new business models and service scenarios will be created in the Cloud. Huge opportunities for technology providers, particularly in supporting activities, are to be seen and will contribute to more Cloud adoption in business and industry. Support activities are vital for enabling user created services, developing new services, and leading the Cloud sustainability process. Finally, we believe that as the Cloud develops in both technical and business aspects, more participation and user-oriented applications will be realized in the Cloud service market.

Annex 1. Case studies

Disclaimer: Please note that due to space limitations, The following cases are just a random sample of the original list of cases studied. The full list is available on request.

Product	Provider	Description and Range of Service	Service / Product	Price / Free	Value chain layer -sub	Website	Comments
Platform Accelerate and Platform Manager	platform	Tools for High Performance Computing (HPC) management software solutions and accelerate compute and data intensive applications and manage cluster and Grid systems	Product and services	Priced	End-to-end solutions that Span over Divers level layers from primary to Cloud and business support layer	http://www.platform.com	Underplaying partnerships and Alliance with other service providers
AppLogic Grid operating system	3tera	Software, on demand and Utility computing	Product and services	Priced	Primary, Cloud and business support activities	http://www.3tera.com	Example of integrating service companies
EnFuzion	axceleon	Resource Management and Workflow Automation, Smart File Transfer and Network Utilization, job scheduler and monitoring, Intuitive GUI,, Extensive Administrative and Reporting Tools.	Product and services	Priced	Primary and Cloud support layers.- applications and software packages with horizontal and vertical integration	http://www.axceleon.com/products.html	–
JBoss Enterprise Application Platform, and Red Hat Enterprise MRG, JBoss Enterprise Middleware and cluster suite	RedHat	Wide range of Grid application and data managements systems. Cloud computing and Grid computing services.	Products and service	Priced	Primary and Cloud support layer activites: SW, MW and applications and consultation	http://www.redhat.com/products/	–
Gridworks, PBS and Portable Batch System	Altair	On-Demand Computing Software Environment, Workload management, and Resource-based scheduling	Products	Priced	Primary – Middle ware – scheduling, management,	http://www.pbsgridworks.com	Support by Multiple hardware and software -partner with intel, windows, hp, Ibm AND OTHERS

Hyperworks	Altair	Optimization, Modeling and Assembly, Virtual Manufacturing, FEA Solvers, Process Automation, Visualization and Reporting, Data Management A Complete Enterprise Simulation Suite	Product and Service	Priced	1- Primary -middleware -management 2-Gid support layer -operators and solution providers	http://www.altairhyperworks.com	Partner and supporter of multiple platforms
HiQube	Altair	Software package for high-performance business intelligence (BI)	software Product (as well Pay-For-What-You-Use" service)	Priced	Cloud support layer – software Business support layer-application	http://www.hiqube.com/	–
GridServer, FabricServer, VersaVision, Federato, DART	Data Synapse	Dynamic manage, and optimize scalable enterprise-class application services as well as metering and runtime control of applications across physical and virtual infrastructure	Products and services	Priced	Primary layer – SW applications and Cloud support layer.	http://www.datasynapse.com/	–
1- Sun Grid Engine, 2- Sun Blade, 3- Sun VDI Software 4- Sun Cloud	Sun	Diverise services and prodcuts including Hardware Servers and Software package for Grid management and middleware resource sharing as well as Multi-clustering Accounting, Reporting Advance Reservation Applications.	Product	Priced	Primary activities: hardware, resource management & Cloud Support activities: Storage Software, applications, accounting	http://www.sun.com/software/sge	Wide range of Grid applications
AppZero's software	AppZero	Wide set of applications for cloud computing, and server-side application virtualization	Product	Priced	Primary: MW, SW and Cloud support activities	http://www.appzero.com	–
Aneka	Manjra soft	Enterprise Grid/cloud computing systems	Product	Priced	Primary – middleware – Cloud support – financial and economic support	http://www.manjrasoft.com	–

SIMtone SNAP	SIMtone Cooperation	Cloud computing: end-to-end Universal Cloud-Computing Platform that allows network operators, service providers and businesses of any size to deliver all types of computing as services to any device	Service DaaS/ SaaS)	Priced	End-to-end cross layer primary, Cloud and support layer services.	http://www.simtone.net/snapbook.htm	–
SIMtone's HPC ASPEED	SIMtone Cooperation	Application and software	product	Priced	Primary activities – package for resource management and – Cloud support activities	http://www.simtone.net/	–
XtremWeb	IN2P3 (CNRS), INRIA and University PAris XI	Middleware Software	Product	Free	Primary activates – Middleware	http://www.xtremweb.net/	
Oracle Grid Products	Oracle	Database management, middleware, VM and enterprize solution	Product and services	Priced	Primary layer- MW – data management, Grid applications, and Cloud support layer	http://www.oracle.com/technologies/grid/grid_products.html	
asiGRID BETA	Andre scavage Soft ware Inc.	Service-Oriented Infrastructure, to manages deployment, routing, scalability, fault-tolerance, security, monitoring, and upgrading of Gird	Product	Priced	Primary and Cloud support layer: application	http://www.gridnow.com/	
Unified Computing System	Cisco	network, storage, server and virtualization	Product and Service	Priced	Primary and Cloud support activities	http://www.cisco.com	
VMware	EMC2	Virtualization	Products and service	Priced	Cloud support activates	http://www.emc.com/services/index.htm	Provide a wide set of applications, consulting and implantation service that support the Cloud
Vine Toolkit and GridSphere 3	Grid Sphere project	Wide range of Grid applications	Product and services	Free	Cloud support activities	http://www.gridsphere.org	Many applications and service for facilitating Cloud business adoption

Dell cloud computing solution	Dell	Hardare and Applications including Designed -TO-Order Project Management and data center customized services	Service and products	Priced	End-to-end Cross layer Primary, Cloud and business. Vertical value.	http://www.dell.com/cloudcomputing	A number of other examples include: Nimbus, Cyclecomputing, etc ...
g-Eclipse	g-Eclipse consortium	Middleware, applaictions (Migrating Desktop, the GridBench suite, the Grid Visualisation Kernel GVK) framework	Product	Free	Primary activities-middleware, SW and management	http://www.geclipse.org	–
3PAR Utility Storage	3PAR	utility storage, and enterprise IT utility services	Service	Priced	End-to-end Cross layer Primary, Cloud and business. Vertical value.	http://www.3par.com/about_us_overview.html	–
The P-GRADE Grid Portal	Computer and Automation Research Institute Hungarian Academic of Sciences	web based environment for the development, execution and monitoring of workflows	Product	Free	Primary activists – SW applications and Cloud support	http://www.p-grade.hu/	Provide support for different workflow tools. Other examples in this catagory include: Gridsphere, OGS portal, etc ...
VIRTERA's vSpectrum	VIR TERA	Provide professional services and consulting firm delivering virtualization technology solutions and services.	Product and services	Priced	Cloud support activities and business support activities	http://www.virteratech.com/	
Globus Toolkit	Globus Alliance	Middleware, solutions and Grid systems and applications	Product	Free	Primary and Cloud support activities	http://www.globus.org	One of the most accesspted and widely adopted middleware and software Standard industry solutions
Amazon Elastic Compute Cloud (Amazon EC2)	Amazon	Hardware and software services as Application Hosting, Backup and Storage, Content Delivery, High Performance Computing, Media Hosting, and On-Demand Workforce. see Amazon Web Services	service	Priced	End-to-end Cross layer Primary, Cloud and business. Vertical value.	http://aws.amazon.com/ec2/	Multiple services for different customer classes

Grid Dynamics solutions	Grid Dynamics	Provide service consulting and development strategies, technology selection and implementation.	Service	Priced	Cloud support layer – consultants	http://www.griddynamics.com	–
COMPOSITE solutions	Compo site software	Provide discovery, Federate and deliver real-time information without physical replication and consolidation. Data Virtualization Solutions	Products and service	Priced	Cloud support activities layer – added value	http://www.compositesw.com/	–
UniCluster and Grid MP products	Univa UD Inc.	Diverse product that address many if not all address of Cloud. Served as stand alone or suit of solution and products as UniCluster, UniPortal,Grid MP, Grid MP, DataCatalyst, and UniSight.	Product and services	Priced	Primary, Cloud and business, support activities	http://www.univaud.com/	Wide range of partnership with Technology Partners, Enterprise Partners, Resellers Systems Integrators
ActiveVOS	Active Endpoints	Provide standards-based visual orchestration system SOA-based process orchestration and business process management (BPM) system.	Product	Priced	Cloud support layer: applications and developers	http://www.activevos.com/	–
Cloud Computing solutions from IBM	IBM	Hardarwe and software solutions	Mainly Products, services also available	Priced	Primary layer activists and Cloud support activities	http://www-03.ibm.com/grid/	target business SME's and enterprises –
cloud-based virtual lab	Skytap	Diverse solutions for cloud computing that include applications that support and integrate with other like Xen	Product and services	Priced	Primary layer activists and Cloud support activities	http://www.skytap.com	–
OGSA-DAI	EPCC at the University of Edinburgh and at NeSC	Data management middleware	Product	Free	Primary layer activities – SW application	http://www.ogsadai.org.uk	–

MOSIX, TXTDemand, and Video Recognition Environment	SORMA (Self-Organizing ICT Resource Management)	Provide Grid management and operating system. In addition to a demand forecasting application and a motion tracking application that creates a graphic real-time presentation.	Product	Priced and Free	Primary and Cloud support activities: SLA based Resource management	http://sorma-project.org	–
SLA management framework Technologies	SLA@SOI (Empowering the Service Economy with SLA-aware Infrastructures)	Provide standard interface for e-contracting platform between service consumers and providers	Product	Free	Primary and Cloud Support activities: SLA management	http://sla-at-soi.eu	–
RESERVOIR Technologies	RESERVOIR (Resources and Services Virtualization without Boundaries)	Provide Infrastructure for Service Delivery services that seek to leverage the diversity factor and achieve economies of scale (Virtual Execution Environment)	Services	Priced	End-to-end services + Cloud support activities and business support activities: infrastructure Virtualization services.	www.reservoir-fp7.eu	–

References

[1] Next Generation Grids Expert Group. Future for European Grids: GRIDs and Service Oriented Knowledge Utilities. Third Report.(January 2006). ftp://ftp.cordis.europa.eu/pub/ist/docs/grids/ngg3_eg_final.pdf.

[2] European Commission. 2006. From Grids to Service-Oriented Knowledge Utilities A critical infrastructure for business and the citizen in the knowledge society. SOKU brochure. Office for Official Publications of the European Communities. Luxembourg. ftp://ftp.cordis.europa.eu/pub/ist/docs/grids/soku-brochure_en.pdf.

[3] European Commission. 2003. Grids Challenges and opportunities for business & industry. European Commission Information Society Directorate-General. Final Report of the European Study Contract No. 30-CE-065970/ 00-56. ftp://ftp.cordis.europa.eu/pub/ist/docs/grids/ist_grids_brochure_oct_2003.pdf.

[4] M. E. Porter. 1998. On Competition. A Harvard Business Review Book. Boston.

[5] M. E. Porter. 1985. Competitive Advantage. Creating and Sustaining Superior Performance. P33. Free Press. New York.

[6] Wikipedia. 2007. Website. Accessed in (November, 24, 2007). http://www.wikipedia.org.

[7] R. Breite and H. Vanharanta. 2002. Value Chain Management in the Internet Environment. In: Twelfth International Working Seminar on Production Economics. Igls/Innsbruck, Austria, Volume 2. (February 18–22, 2002) http://www2.ipe.liu.se/rwg/igls/igls2002/Paper044.pdf.

[8] V. Allee. 2003. The Future of Knowledge: Increasing Prosperity through Value Networks. Butterworth-Heinemann.

[9] F. K. Pil and M. Holweg. 2006. Evolving from value chain to value grid. In: MIT Sloan Management Review, 47(4): 72–80.

[10] K. Stanoevska-Slabeva, C. F. Talamanca, G. A. Thanos and C. Zsigri. 2007. Development of a Generic Value Chain for the Grid Industry. In: Grid Economics and Business Models. Volume 4685/2007, Pages 44–57. Springer Berlin, Heidelberg.

[11] Gridpedia, 2008, Website, accessed 21/12/2008. http://www.gridipedia.eu/index.php?id=690.

[12] S. Forge and C. Blackmann. 2006. Commercial Exploitation of Grid Technologies and Services – Drivers and barriers, Business Models and Impacts of Using Free and Open Source Licensing Schemas. Final Report of the European. Study Contract No. 30-CE-065970 /00-56. (November 2006).

[13] H. J. Lee and C. S. Leem. 2005. A study on value chain in a ubiquitous computing environment. In: Computational Science and Its Applications ICCSA 2005. Volume 3483/2005. pp. 113–121. Springer Berlin, Heidelberg.

[14] J. Altmann, M. Ion and A. Bany Mohammed. 2007. Taxonomy of Grid Business Models". In: Grid Economics and Business Models. Volume 4685/2007, Pages 29–43. Springer Berlin, Heidelberg.

[15] Globus project, Website, accessed 21/12/2008. available at http://www.globus.org.

[16] I. Foster and C. Kesselman, 1999. Globus: A Toolkit-Based Grid Architecture. in The Grid, Blueprint for a Future Computing Infrastructure", I. Foster, 196 and C. Kesselman, Editors, Morgan Kaufmann, San Francisco, pp. 259–278.

[17] A. Grimshaw, A. Ferrari, F. Knabe and M. Humphrey. 1999. Legion: An operating system for wide-area computing. IEEE Computer, 32((5)).

[18] Enabling Grid for E-sciencE (EGEE), 2008, Website, accessed 21/12/2008 www.eu-egee.org.

[19] gLite 3.0.0, home page, 2007. http://www.glite.org.

[20] D. W. Erwin and D. F. Snelling. 2001.UNICORE: A Grid Computing Environment" in R. Sakellariou et al. (Eds.): Euro-Par 2001, LNCS 2150, pp. 825–834.

[21] J. Altmann and D. J. Veit. 2007. Grid Economics and Business Models. LNCS 4685, ISBN 0302–9743. Springer-Verlag, Berlin, Germany. http://it.i-u.de/schools/altmann/gecon.

[22] Amazon. 2007. Web Services. Accessed 21/9/2007. http://www.amazon.com/gp/browse.html?node=201590011.

[23] Sun Grid. 2007. Website., Accessed 21/9/2007. http://www.sun.com/service/sungrid.

[24] D. Harris. 2007. My Day at SC07. In Grid today. Vol. 6, No. 46. (November 19, 2007).

[25] S. Pena. 2007. Outsourcing of Grid Computing. A Project Report, Department of Computer Science, Louisiana State University. (May 23, 2007). http://www.cct.lsu.edu/~gallen/Students/Pena_2007.pdf.

[26] La Grid.2007. Website. Accessed 25/10/2007. http://latinamericangrid.org.

[27] D-Grid. 2007. Website. Accessed 25/10/2007.www.d-grid.de.

[28] Seti@home. 2007. Website. Accessed 27/10/2007. http://setiathome.berkeley.edu.

[29] Einstein@home. 2007. Website. Accessed 27/10/2007. http://einstein.phys.uwm.edu.

[30] rosetta@home. 2007. Website. Accessed. 27/10/2007. http://boinc.bakerlab.org/rosetta.

[31] M. Hergert and D. Morris 1989. Accounting data for value chain analysis. Strategic Management Journal, Vol. 10, p. 175.

[32] I. Foster, H. Kishimoto, A. Savva, D. Berry, A. Djaoui, A. Grimshaw, B. Horn, F. Maciel, F. Siebenlist, R. Subramaniam, J. Treadwell and J. V. Reich. 2005. The Open Grid Services Architecture, Version 1.0, Open Grid Services Architecture WG, Global Grid Forum.

[33] Gemlca. 2008. Website. Accessed. 1/15/2008. http://www.cpc.wmin.ac.uk/cpcsite/index.php/Gemlca.

[34] E. Seidel, G. Allen, A. Merzky and J. Nabrzyski. 2002. GridLabA Grid application toolkit and testbed, Future Generation Computer Systems 18 (8), pp. 1143–1153.

[35] J. Spohrer, L. C. Anderson, N. J. Pass, T. Ager and D. Gruhl. 2008. Service Science. J Grid Computing(6). PP. 313–324.

[36] M. Risch and J. Altmann. 2008.Cost Analysis of Current Grids and its Implications for Future Grid Markets. GECON 2008, Workshop on Grid Economics and Business Models, Springer LNCS, Las Palmas, Spain.

[37] J. Altmann, C. Courcoubetis, G. D. Stamoulis, M. Dramitinos, T. Rayna, M. Risch and C. Bannink. 2008. GridEcon – A Market Place for Computing Resources. GECON 2008, Workshop on Grid Economics and Business Models, Springer LNCS, Las Palmas, Spain.

[38] J. Altmann. 2000. A Reference Model of Internet Service Provider Businesses. ICTEC2000, 3rd International Conference on Telecommunication and Electronic Commerce, Dallas, Texas, USA.

[39] J. Altmann and I. Constantiou. 2004. Towards a Profitable ISP Business in a Competitive Environment. book chapter titled for the book Social and Economic Transformation in the Digital Era. IDEA Group Publishing, USA, 2004.

[40] J. Altmann, D. Neumann and T. Fahringer. 2008. Grid Economics and Business Models. LNCS, Springer-Verlag, Berlin, Heidelberg, Germany.

[41] Xen, 2009. Web site. Accessed. 5/15/2009. http://www.xen.org.

Ashraf Bany Mohammed
Technology Management, Economics and Policy Program
Department of Industrial and Management Engineering
College of Engineering
Seoul National University
San 56-1, Silim-dong
Gwanak-Gu, Seoul 151-742
South-Korea
e-mail: ashraf@temep.snu.ac.kr

Jörn Altmann
Technology Management, Economics and Policy Program
Department of Industrial and Management Engineering
College of Engineering
Seoul National University
San 56-1, Silim-dong
Gwanak-Gu, Seoul 151-742
South-Korea
e-mail: jorn.altmann@acm.org

Junseok Hwang
Technology Management, Economics and Policy Program
Department of Industrial and Management Engineering
College of Engineering
Seoul National University
San 56-1, Silim-dong
Gwanak-Gu, Seoul 151-742
South-Korea
e-mail: junhwang@snu.ac.kr

Economic Models and Algorithms for Distributed Systems, 209–220
Book Series: Autonomic Systems

A Model for Determining the Optimal Capacity Investment for Utility Computing

In Lee

Abstract. Utility computing has emerged as a new IT model for future computing and storage resource management for enterprises. Utility computing has drawn attention from enterprise customers who seek to reduce upfront IT investment and enhance computing agility. Major IT service providers envisioned that the commoditization and standardization of IT resources would usher in the shift of the IT paradigm towards utility computing. In this paper, a decision model is presented for determining the optimal capacity of a utility computing which maximizes the total profit for a utility computing service provider. The model considers both capacity investment cost and demand level, and derives closed form solutions for the investment.

Mathematics Subject Classification (2000). Primary 68Q10; secondery 68Q85.

Keywords. Cloud computing, value chain, business models, grid computing, service oriented computing, value networks, software-as-a-service, grid economics, services, service sciences.

1. Introduction

Utility computing is a technology for computing and storage resources in which computing resources to users are available on an as-needed basis, similar to electricity and gas. The utility computing market has already begun to take shape with a number of service providers that offer a range of service options. Leading firms in the computing industry such as IBM, Sun Microsystems, and Hewlett-Packard have pursued their own versions of utility computing services, the success of which still remains to be seen. Other companies in the computing industry also have vast capabilities of technical staff, computers, data storage, and networks that are frequently underutilized and can readily sell the underutilized IT resources to other enterprise customers. Traditional non-IT organizations are also entering into

this relatively new business area with the expectation of new business potential. For example, Amazon started to sell computing and storage resources and is currently heavily investing in the resource expansion and marketing. Amazon's entry into the utility computing market showcases that the IT resources are moving into the direction of commodity products and services, and that the market entry barrier into utility computing would be as low as e-commerce given the technology is mature.

Resource underutilization is typical in the firms. Individual computing resources are dedicated to specific applications in the anticipation of peak-load demand over multi-year life cycle of the resources. Utility computing can potentially reduce the financial and technological risks involved in direct ownership of technology for companies whose underutilization wastes precious corporate resources. The value proposition of utility computing has evolved from a cost-oriented approach whose purpose is to drive down the total cost of computing to a strategy-oriented approach in which computing resources are economically utilized in meeting computing demands and responding to quick changes in the business environment. Carr [3] argued that because IT is ubiquitous, the greatest IT risk is overspending, putting companies at a cost disadvantage. He further suggested that companies need to direct savings from IT investment into other areas where enterprises can achieve core competence.

Capacity planning is a crucial business decision for utility computing service providers. The demand for high performance utility computing capacity is likely to increase in the future by customers who seek high performance computing solutions due to ever increasing data volumes. The investment decision requires the determination of optimal capacity level for computing demand. There are always two possibilities in the investment: overcapacity and under-capacity. In a manufacturing environment, under-capacity problem may be solved with overtime, additional shifts, and inventory stocks. While over-capacity in computing wastes financial resources and contribute to lower profits, under-capacity computing resources are detrimental to service providers and customers due to the time-sensitive nature of the service and loss of sales opportunities. For example, timeliness of computing would be critical in a financial industry where financial data has a high time-based value. In light of the importance of the right IT investment decisions and the lack of studies in the IT investment area, our paper proposes a decision model of the utility computing capacity investment for service providers. The remainder of this paper includes a literature review on utility computing, the decision model with illustration the analysis of utility computing investment utilizing the decision model, and conclusion.

2. Literature review

Most of the research in utility computing has focused on business models, pricing methods, infrastructure development, resource management, scheduling, and security management. In this section, we review utility computing from diverse

perspectives. First, we review various definitions of utility computing proposed in the literature. Ross and Westerman [11] define utility computing as a collection of technologies and business practices that enables computing to be delivered seamlessly and reliably across multiple computers. IBM defines utility computing as a service provisioning model in which a service provider makes computing resources and infrastructure management available to the customer as needed, and charges them for specific usage rather than a flat rate [12]. The utility model seeks to maximize the efficient use of resources and/or minimize associated costs. Utility computing is a paradigm where shared infrastructure can be provided on demand to multiple applications [8]. Sun Microsystems defines utility computing as a business model for computing in which resources (CPU power, storage space, etc.) are made available to the user on an as-needed basis. Key to these definitions is the notion of on-demand IT resource provision and sharing of resources. Technologies behind utility computing include the virtualization, grid computing, web services, and autonomic computing.

A number of researchers presented a utility computing business model [2, 10]. Due to a lack of clear understanding of a business model itself, there seems to be a divergence in the discussion of the utility computing business model. Requirements of successful utility computing services include affordability, accessibility, scalability, reliability, security, and usability [10]. A number of services were offered to serve diverse customer needs. In the following, we summarize these services:

Data-oriented services

offer storage space and bandwidth based on pay-per-use or storage space. For instance, a data center with a global information infrastructure can be equipped with thousands of high performance servers and a comparable amount of storage capacity to allow customers to back up data and store files remotely.

Computation-oriented services

offer computing resources usually on pay-per-use basis, well-suited for computation-intensive tasks such as product design, chemical analysis, and movie rendering. Computation-oriented services are targeted for customers who need to buy on-demand computing capacity to handle irregular peak workloads. Computation-oriented applications are widely used in energy, bioinformatics, aerospace, insurance, and aerospace industries.

Application-oriented services

offer application software together with the necessary computing infrastructure on which to run it. Common examples of the applications offered include Customer

Relationship Management (CRM) Systems, Database Management Systems, and Payroll Systems.

Consulting services

provide expertise to customers who are considering switching their computing from internal IT resources to utility computing. They help customers achieve their business objectives by leveraging the value of utility computing services.

The success of utility computing depends largely on good capacity planning and customer management. The realization of the full potential of utility computing services depends largely on the computing capability to satisfy the demand of its enterprise customers. Typically, enterprise customers demand three things from utility computing:

1. Quality of service
2. Speedy response to computing requests
3. Competitive price

In the computing industry, capacity planning, the calculation of the number of servers or storage units needed to serve future computing need, is difficult because of sensitivity to computing job mix and uncertainty in demand. Planning for a single customer can result in a large gap between planned capacity and actual capacity needs when the realized computing requests turn out differently from the one planned. Stability of utilization is achieved through the multiplexing of different computing needs over time. It is important to keep a satisfactory response time of requests, high throughput, and an acceptable service quality. Given the general lack of understanding of computing resources and customers service needs, the computing service providers may invest too much or too little for utility computing. Capacity plan must ensure that the service providers deliver quality computing services to customers at an acceptable response speed. If capacity is over-utilized, the response time gets slow, or no action is taken for the capacity planning, then customers may switch to competitors to find better service, quality, and speed.

When we consider investment in capacity expansion of utility computing, it is important to consider how customer demand is affected by the services of a utility computing. The decision would be the optimal capacity investment that maximizes profit considering the investment cost and customer's responses to the service offered with the invested capacity. While substantial studies have been conducted in the area of network scheduling and resource management, approaches which can be implemented as managerial tools for investment decisions are generally lacking. Discussions on a decision model are presented in the next section.

3. A decision model

In this section, we present a model for capacity and demand planning that involves both pricing and investment cost. This mechanism is the first attempt

in both adding premium pricing and demand considerations into the capacity investment. Our methodology applies optimization techniques with decision calculus approaches offering sufficient flexibility for applied contexts. The difference between capacity and demand is used as a proxy for service quality to capture customers' response the quality of the service. That is, the larger the difference is, the faster the speed of computing, and therefore the higher customer satisfaction. In this paper, we capture customer's response to service using a decision calculus model widely used in marketing research and practices [1, 5, 6, 9]. The selection of Little's 1970 paper on decision calculus was selected as one of the most influential in the first 50 years of Management Science (2004). Since price and speed of the service are the two most important attributes of any service provisions, a decision calculus model was incorporated into the investment decision model to measure the quality of the computing service and to convert the superior service into a premium service. While other attributes are important and affect the capacity investment decision, we consider only the speed aspect of service for optimal investment decision. Although other attributes are important, we believe that their impact on the capacity investment decision is secondary when compared to speed of computing for a unit of service.

The decision model should be calibrated by examining subjective judgments on model parameters and outcomes from the decision variables (e.g., profit and capacity investment) under a variety of hypothetical scenarios (e.g., demand level, customers' sensitivity to the service quality). Once the model linking parameters to capacity decision variables has been calibrated, an optimal capacity level is derived. First, we introduce a nomenclature used throughout this paper and discuss a model formulation.

Nomenclature
α: Minimum coefficient of premium service
β: Maximum coefficient of premium service
C: Capacity investment in service units
D: Demand in service units
H: Maximum incremental capacity investment
L: Minimum capacity investment
κ: Unit service price
γ: Unit service cost
V: Investment cost for capacity investment of C

4. Base model

In this section, we assume that V is a function of a capacity investment of C. The valid range of the premium service for service quality should depend on the specific services they are offering. The premium service index indicates that the premium service linearly decreases as the difference (C-D) between capacity and demand narrows. To estimate the functional form of premium service, we adopt

an approach successfully used in many decision calculus models [4, 5, 7]. In this paper, α represents minimum coefficients of premium, and β represents maximum coefficients of premium service. For the construction of reasonable functional form, managers need to estimate the difference (C-D) which is required to maintain the base price (i.e., no premium service).

$$\max NP = \left[\alpha + \frac{(\beta - \alpha)(C - D)}{H} + 1\right]\kappa D - \gamma D - V$$
$$= \alpha\kappa D + \frac{(\beta - \alpha) C\kappa D}{H} - \frac{(\beta - \alpha) kD^2}{H} + \kappa D - \gamma D - V\,. \tag{4.1}$$

The objective of Equation 4.1 is to maximize the total net profit from the investment decision.

$$\text{Premium Service Quality, } \Omega = \left[\alpha + \frac{(\beta - \alpha)(C - D)}{H} + 1\right]. \tag{4.2}$$

As the difference between C and D gets larger, the premium service quality index gets larger. The maximum possible value for the premium service quality index is 1 + β and the minimum possible value for the premium service quality index is 1 + α. H is the maximum incremental capacity investment, C is the capacity invested, and D is the demand for utility computing. Therefore, when C is the same as D, the minimum possible value for the premium service quality index of 1+ α is obtained. As an illustration, suppose α = -0.25, β = 0.45, H = 10000 L = 3000 C = 8000 and D = 4000, then premium service index, Ω, is 1.030. When C is 7572, Ω, is 1.000. In the following procedures, we identify a closed form solution for the optimal capacity investment of C^*. If the first derivative of NP is taken with regard to V and set to zero, and solved, then the result is

$$\frac{\partial NP}{\partial V} = \frac{(\beta - \alpha)\kappa DC'}{H} - 1 \tag{4.3}$$
$$\frac{(\beta - \alpha)\kappa DC'}{H} - 1 = 0 \tag{4.4}$$
$$\frac{\partial C}{\partial V} = \frac{H}{(\beta - \alpha)\kappa D}\,. \tag{4.5}$$

Let's assume that C is a limited growth function with base e used to estimate the capacity increase for investment.

$$C = H\left(1 - e^{-\lambda V}\right) + L\,, \quad V \geq 0 \tag{4.6}$$
$$C = H - He^{-\lambda V} + L\,. \tag{4.7}$$

The first derivative of C with regard to V is

$$\frac{\partial C}{\partial V} = \lambda H e^{-\lambda V} = \lambda \left(H + L - C\right) > 0 \tag{4.8}$$

$$\frac{H}{\left(\beta - \alpha\right)\kappa D} = \lambda \left(H + L - C\right) . \tag{4.9}$$

If D is known, C^* is derived by the following procedures.

$$\lambda C = \lambda H + \lambda L - \frac{H}{\left(\beta - \alpha\right)\kappa D} \tag{4.10}$$

$$C^* = \frac{\lambda H + \lambda L - \frac{H}{(\beta - \alpha)\kappa D}}{\lambda} . \tag{4.11}$$

Since $C^* = H - He^{-\lambda V} + L$

$$V^* = \frac{\left(\ln \frac{(H + L - C^*)}{H}\right)}{-\lambda} . \tag{4.12}$$

Next, an illustration of the model operation is given below. Let's assume the following:

α = -0.25; β = 0.45; H = 10,000; L = 3,000; λ = 0.00003; κ = 120; γ = 80. If D = 4,000, then the optimal capacity, C^* is 12007.93651 and optimal investment, V^* is 77018.44209. Finally, NP^* is calculated as 232048.2246.

5. Simultaneous optimization of capacity and demand

In the previous section, we considered the utility computing capacity decision making to obtain a maximum profit under the fixed demand. In this section, we develop closed form solutions for both capacity and demand. We assume the service providers can decide the demand level through marketing efforts. First, we rearrange the NP in terms of D.

$$NP = -\frac{\left(\beta - \alpha\right)k}{H} D^2 + \left(\frac{\left(\beta - \alpha\right) C\kappa}{H} + \alpha\kappa + \kappa - \gamma\right) D - V \tag{5.1}$$

Let $\frac{(\beta - \alpha)}{H} = \varpi$. Then $NP = -\kappa\varpi D^2 + \left(\kappa\varpi C + \kappa\alpha + \kappa - \gamma\right) D - V$. (5.2)

If the first derivative of NP is taken with regard to D and set to zero, and solved, then the result is

$$\frac{\partial NP}{\partial D} = -2\kappa\varpi D + (\kappa\varpi C + \kappa\alpha + \kappa - \gamma) \tag{5.3}$$

$$\text{Let } D = \frac{(\kappa\varpi C + \kappa\alpha + \kappa - \gamma)}{2\varpi\kappa}$$

$$\text{Let } \frac{(\kappa\varpi C + \kappa\alpha + \kappa - \gamma)}{2\varpi\kappa} = \frac{1}{\varpi\lambda\kappa\,(H + L - C)}$$

$$\text{Let } \lambda\kappa = \eta\,. \text{ Then } \frac{1}{\varpi\lambda\kappa H + \varpi\lambda\kappa L - \varpi\lambda\kappa C} = \frac{1}{\varpi\eta H + \varpi\eta\ L - \varpi\eta C}$$

$$\text{Let } \kappa\alpha + \kappa - \gamma = \rho\,. \text{ Then } \frac{(\kappa\varpi C + \kappa\alpha + \kappa - \gamma)}{2\varpi\kappa} = \frac{(\kappa\varpi C + \rho)}{2\varpi k}$$

$$C^* = \frac{(\kappa\varpi\eta L + \kappa\varpi\eta H - \rho\eta)}{2\kappa\varpi\eta} \pm \pm \frac{\sqrt{\left(-\left(\kappa\varpi\eta L + \kappa\varpi\eta H - \rho\eta\right)\right)^2 + 4\kappa\varpi\eta\,(\rho\eta L + \rho\eta H - 2k)}}{2\kappa\varpi\eta} \tag{5.4}$$

$$D^* = \frac{\left(\frac{\kappa(\beta-\alpha)C^*}{H} + \kappa\alpha + \kappa - \gamma\right)}{\frac{2(\beta-\alpha)\kappa}{H}}\,. \tag{5.5}$$

Next, an illustration of the model operation is given below. Let's assume the following:

α = -0.25; β = 0.45; H = 10,000; L = 3,000; λ = 0.00003; κ = 120; γ = 80. Applying Equations (16) and (17), the optimal capacity, C^*, is 12416.743, the optimal investment, V^*, is 94723.739, and the optimal demand, D^*, is 6803.6095. Finally, NP^* is calculated as 294104.72.

6. Analysis of model behaviors

In this section, we analyze the relationship between the capacity decision, the demand decision, and model variables. The first experiment considers a set of possible, discrete demand scenarios and determines the optimal capacity and investment cost related to the capacity to maximize the total profit. Figure 1 shows that as the demand level increases, the optimal capacity increases, but at a higher demand level, the growth of the capacity becomes slow. The result is attributable to the fact that the capacity investment function is exponential as shown in Equation (4.6).

Figure 2 shows that as the unit price increases, both the capacity and demand should increase. However, the rate of increase in the demand is faster than the rate of increase in the capacity. This is attributable to the fact that as the unit

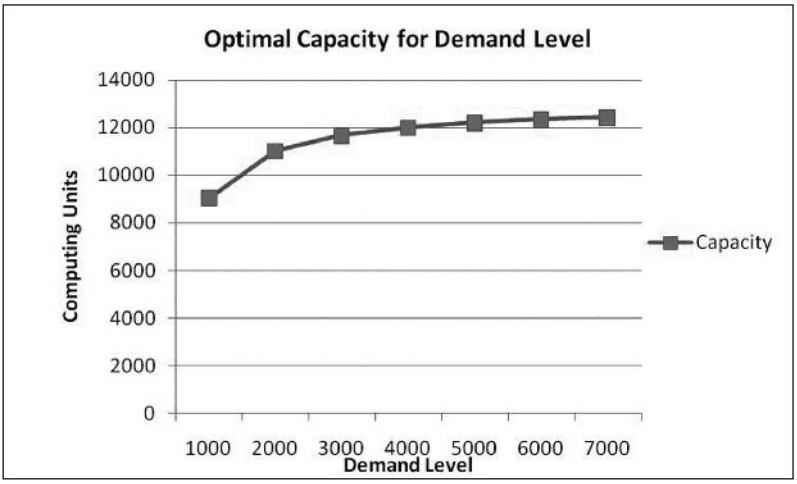

FIGURE 1. Optimal capacity for demand level.

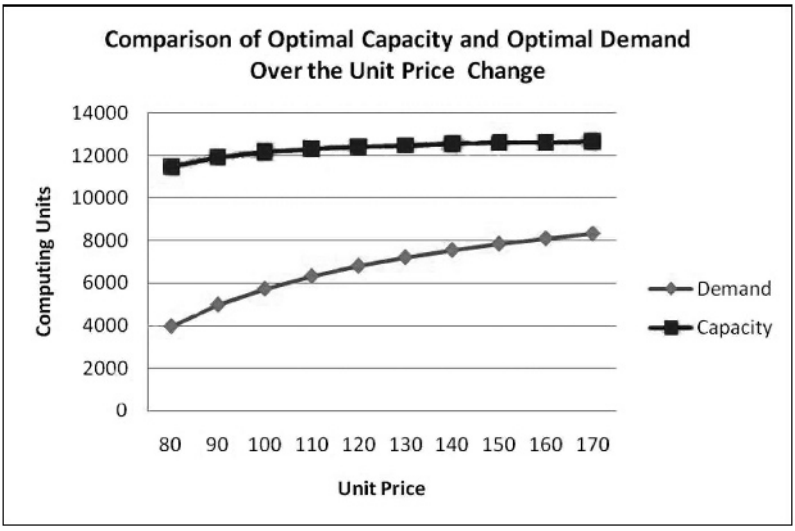

FIGURE 2. Comparison of optimal capacity and optimal demand over the unit price change.

profit (unit price ' unit cost) increases, the benefit of a higher price outweighs the premium service obtained from the fast response time.

Figure 3 shows that as the unit cost increases, both the capacity and demand should decrease. However, the rate of decrease in the demand is faster than the rate of decrease in the capacity. This is attributable to the fact that as the unit

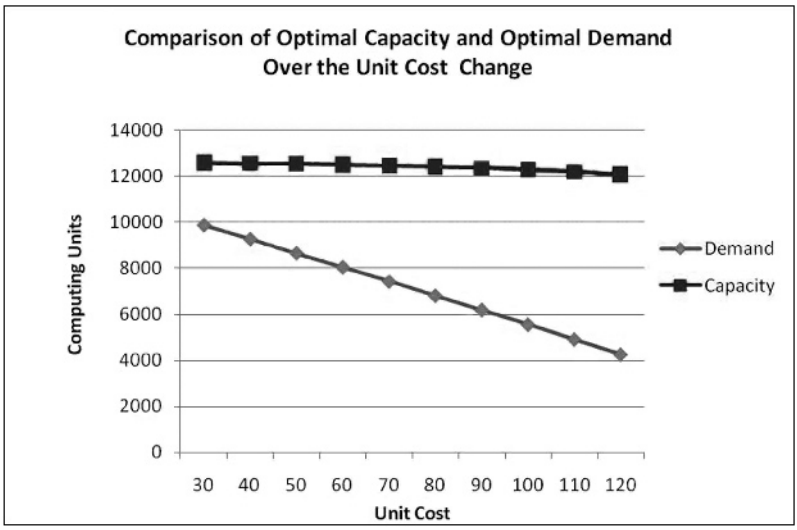

FIGURE 3. Comparison of optimal capacity and optimal demand over the unit cost change.

profit (unit price ' unit cost) decreases, the benefit of premium service obtained from the fast response time outweighs the cost of capacity investment.

Figure 4 shows that as the maximum coeffient of the premimum price increases, the optimal capacity level increases. However, the optimal demand decreases. As the difference between capacity and demand increases, the difference between the optimal demand and optimal capacity becomes wider. The increase of the coeffient of the premimum service means customers are more sensitive to the service quality. Therefore, the increase of the coeffient of the premium service requires the increase of the optimal capacity and the decrease of the optimal demand.

7. Conclusion

In this paper, we presented a capacity planning model for utility computing. We presented a closed form solution for both the optimal capacity and demand. Utility computing service providers can choose to use this model or to develop a more sophisticated simulation model based on the observation of this mathematical model. The successful use of this model requires accurate estimation of the model parameters, and other model estimation tools and techniques. This model also serves as a general guideline for investment and demand management for utility computing service providers.

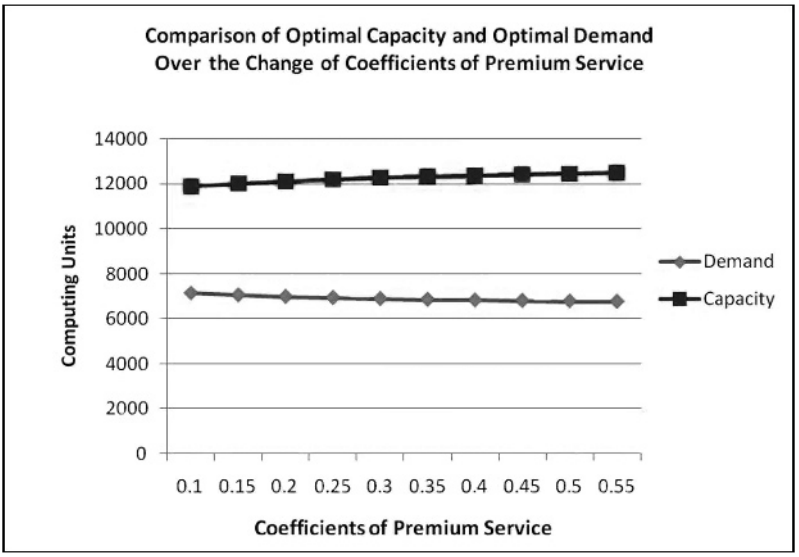

FIGURE 4. Comparison of optimal capacity and optimal demand over the change of coefficients of premium service.

References

[1] D. Chakravarti, A. Mitchell, and R. Staelin, *Judgment based marketing decision models: an experimental investigation of the decision calculus approach* Management Science, **25 (3)** (1979), 251–263.

[2] J. P. Degabriele and D. Pym, *Economic aspects of a utility computing service* http://www.hpl.hp.com/techreports/2007/HPL-2007-101.pdf (2007).

[3] N. Carr, *IT doesn't matter* Harvard Business Review **81 (5)** (2003), 41–49.

[4] J. C. Larrch and V. Srinivasan, *STRATPORT: A decision support system for strategic planning* Journal of Marketing **45** (1981), 39–52.

[5] J. D. C. Little, *Models and managers: The concept of a decision calculus* Management Science **16**, (1970), B466–B485.

[6] J. D. C. Little, *Models and managers: The concept of a decision calculus* Management Science **50 (12)**, Supplement (2004), 1854–1860.

[7] L. M. Lodish, Callplan, *an interactive salesman's call planning system* Management Science **18 (4)**, Part II, (1971), 25–40.

[8] V. Machiraju, C. Bartolini, and F. Casati, *Technologies for business-driven IT management* HPL-2004-101, (2004).

[9] P. M. Parker and M. Sarvary, *Formulating dynamic strategies using decision calculus,* European Journal of Operational Research **98 (3)**, (1997), 542–554.

[10] M. A. Rappa, *The utility business model and the future of computing services* IBM Systems Journal **43 (1)** (2004), available from: http://www.research.ibm.com/journal/sj/431/rappa.html.

[11] J. W. Ross and G. Westerman, *Preparing for utility computing: The role of IT architecture and relationship management* IBM Systems Journal **43 (1)** (2004), available from: http://www.research.ibm.com/journal/sj/431/ross.pdf.

[12] A. Stolvoort, *Utility Computing.* available from: http://www05.ibm.com/nl/events/presentations/Utility_Computing.pdf (2007).

In Lee
Department of Information Systems and Decision Sciences
College of Business and Technology
Western Illinois University
Macomb, IL 61455
USA
e-mail: I-Lee@wiu.edu

Economic Models and Algorithms for Distributed Systems, 221–237
Book Series: Autonomic Systems

A Combinatorial Exchange for Complex Grid Services

Melanie Moßmann, Jochen Stößer, Adam Ouorou, Eric Gourdin, Ruby Krishnaswamy and Dirk Neumann

Abstract. The Grid is a promising concept to solve the dilemma of increasingly complex and demanding applications being confronted with the need for a more efficient and flexible use of existing computer resources. Even though Grid technologies have made progress within the context of large enterprises and academic projects, there has not yet been a widespread adoption by public institutions and small enterprises. One barrier to this adoption is the lack of economic paradigms which support the dynamic and efficient sharing of Grid resources by balancing resource scarcity and idle capacities. Economic algorithms promise to provide a good fit to the Grid's inherent strategic dimension by enabling users to express their valuation for computer resources. At the same time they provide incentives to contribute idle resources to the Grid in return for the market price.

This paper presents a market-based approach for trading complex computational Grid services. The implemented combinatorial exchange aims at maximizing the social welfare of users. At its core, it provides a rich bidding language which is able to represent complex Grid services and simple workflows. The allocation mechanism is evaluated by means of a numerical experiment in order to gain detailed insights into the computational complexity of the underlying allocation problem. Our analysis provides input for the configuration of Grid markets.

Mathematics Subject Classification (2000). Primary 68Q10; secondery 68Q85.

Keywords. Cloud computing, value chain, business models, grid computing, service oriented computing, value networks, software-as-a-service, grid economics, services, service sciences.

1. Introduction

Complex applications and simulations in science and business such as computer-aided engineering or demand and weather forecasts create tremendous demand for computer resources. Many of these applications exhibit fluctuating utilization patterns with few peak loads, which lead to low resource utilization on average. This trend is confronted with limited budgets and the need for a more efficient usage of scarce resources. Grid technology offers a promising way out of this dilemma by enabling the dynamic and efficient sharing of heterogeneous computer resources within organizations as well as across administrative domains [6]. However, despite its promises, there has not yet been a widespread adoption of Grid technologies; only few enterprises and large academic projects leverage the Grid's computational power.

The lack of economic paradigms has been identified as one of the major inhibitors of Grid technology [5]. If computer resources are scarce, a scheduling strategy is needed that decides which resources should be allocated to whom at what time. Classic technical scheduling algorithms such as first-come-first-serve or fair-share are solely built on system-centric metrics. Economic algorithms promise to provide a better fit to the Grid's inter-organizational and thus strategic nature by explicitly taking into account valuations [7]. By enabling resource requesters and providers to report valuations in addition to technical metrics, the scheduling mechanism can consider user preferences for resource allocations. Prices are used to aggregate and disseminate information about the system's status and help to control the dynamic demand and supply, while at the same time providing incentives to contribute idle resources to the Grid.

This paper is structured as follows. We discuss related work and introduce a sample application scenario in Sections 2. and 3., respectively. At the core of this paper, we present a comprehensive combinatorial exchange in Section 4.. In Section 5., the mechanism is evaluated by means of a numerical experiment in order to gain detailed insights into the computational complexity of the underlying allocation problem. Section 6 concludes the paper and points to future research directions.

2. Related work

There are two main streams of research in auction-based scheduling of Grid resources: mechanisms which consider the trading of one type of resource only and mechanisms which account for *dependencies* between multiple Grid resources.

Mechanisms which consider only one resource have been proposed e.g. in [3, 8, 11, 14]. The most prominent mechanism is Market-based Proportional Share where resource requester i with a reported valuation of v_i gets allotted computing time of $v_i / \sum v_j$, i.e. proportional to i's share in the total reported valuation [3]. The main advantage of these mechanisms is their simplicity. Users do not have to

specify complex technical requirements of their computational jobs. Furthermore, due to their computational speed, these mechanisms support real-time applications which require the immediate allocation of computing time. In many settings, however, requesters require a bundle of resources such as computing power, memory, bandwidth and software licenses. On their downside these simple approaches thus lead to inefficient allocations and do not support the advance reservation of resources.

These drawbacks gave rise to mechanisms which approach Grid settings by means of combinatorial exchanges, e.g. [1, 2]. The mechanism presented in [13] is most relevant to the work presented in this paper. In this *Multi-Attribute Combinatorial Exchange* (MACE), users are allowed to request and offer arbitrary bundles of Grid resources and can specify quality attributes and time constraints. Even though complex resource requests may be made, MACE nevertheless does not offer explicit support for workflow execution. Compared to MACE, our approach responds to the specific needs of small enterprises and institutions.

The contribution of this paper is the design of a comprehensive combinatorial Grid exchange, which allows users to exactly specify resource requirements, time constraints, and dependencies between multiple jobs, e.g. several runs within one simulation.

3. Application scenario: Collaborative learning

Grid computing denotes a computing model where distributed and heterogeneous computing resources integrated through middleware and infrastructure software (the so-called "Grid") provide seamless access to applications.

The application scenario focuses on providing dynamic access to Grid resources for educational institutions and small enterprises in contrast to large-scale utility centres, such as Amazon's Elastic Compute Cloud[1] or Sun Microsystem's network.com[2]. Computer supported virtual campuses and collaborative learning have established themselves as a strong complement to classic physical universities as they enable participants to engage in activities that have not been possible due to constraints in time, location etc.

In the context of collaborative learning, participants generate input files to execute simulations that are structured in form of simple workflows. The execution requires processing and storage capacity. That is where the resource allocation mechanism gets involved. It determines the optimal allocation of overall demanded and provided resources. If a request is successful, a list of allocated resources is returned to the participants.

The correlated demand functions for the two resources (processing and storage capacity) dictate a combinatorial auction that allows market participants to offer and request *combinations* of resources. It mitigates the so-called "exposure

[1] http://aws.amazon.com/ec2
[2] http://www.network.com

risk" where consumers may end up with only one resource out of the required combination of resources. The term "job" is used to indicate the total set of resources with their quantity and quality attributes that are required to execute the workflow as well as time attributes. To complete a job, the entire duration needs to be satisfied by meeting the *time consistency* property that forces the allocation to be contiguous in time. A *bundle* refers to a combination of resources that is offered by suppliers.

Consumers indicate preferences among jobs through their valuations, which essentially represent the maximal willingness to pay. The valuation for a job may be greater than that of the sum of the individual resources required to execute the job (super-additive valuations), e.g. $v(CPU) + v(Storage) \leq v(CPU, Storage)$. Consumers express their substitute preferences by submitting multiple exclusive jobs. Analogously suppliers specify the minimal accepted earning through their reservation price for specific bundles.

Resources are traded in discrete time-slots of fixed and uniform length. Agents express availability time frames and consumers additionally the duration of tasks in units of time-slots.

4. A combinatorial exchange

Maximizing social welfare is our main objective as it puts goods in the hands of people who value them most. Generally, users are considered to act strategically in terms of trying to find their optimal (i.e. utility maximizing) strategy. In the following, the bidding specification is introduced in Subsection 4.1, the Winner Determination Problem (WDP) is formulated in Subsection 4.2, and finally the most relevant price mechanisms are discussed in Subsection 4.3.

4.1 Bidding specification

The bidding specification defines the content of communication exchanged between participants and the auction mechanism. A trade-off needs to be done between the choice of a flexible bidding specification enabling expression of arbitrary and complex workflows and simplicity in terms of the resolution mechanism.

Let N be the set of consumer bids[3] consisting each of a collection of jobs $J^n = \{1, \ldots, \overline{j}\}$, which are connected by the XOR[4] operator so that at most one job is accepted $\{\text{Job } 1, \ldots, \text{Job } \overline{j}\}$. The same applies for the set of supplier bids[5] that contains the set of bundles $B^m = \{1, \ldots, \overline{b}\}$. Bids are linked by the OR[6] operator and have the structure $\{\text{Bundle } 1 \vee \cdots \vee \text{Bundle } \overline{b}\}$.

[3]The notations *consumer* and *consumer bid* are used equivalently in this context.
[4]$X \underline{\vee} Y$ means either $\emptyset$, X or Y, but not X and Y
[5]The notations *supplier* and *supplier bid* are used interchangeably in this context.
[6]$X \vee Y$ means either $\emptyset$, X, Y, or X and Y

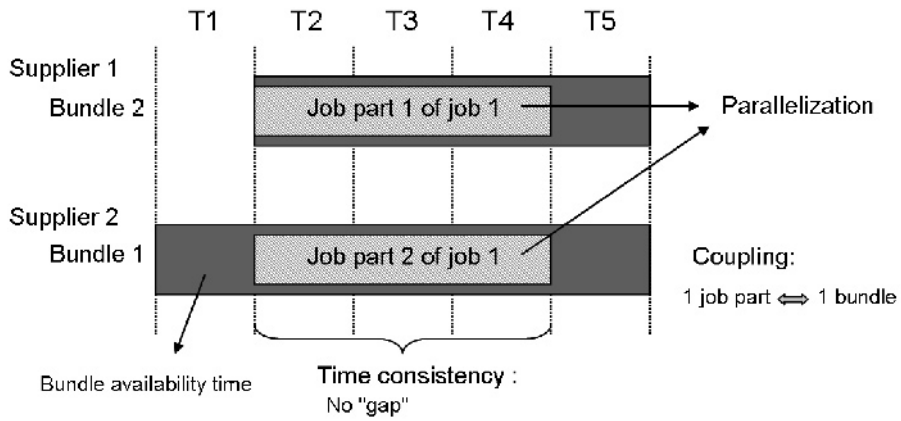

FIGURE 1. Potential allocation with activated parallelization that meets the two central allocation conditions: coupling and time consistency.

Trading on the Internet implies a large number of providers where each offers a small quantity of resources. Thus single bundles may not contain sufficient quantities of resources to satisfy a job, so bundles may need to be aggregated. However, in order to reduce the complexity of the allocation mechanism we do not allow the disaggregation of bundles within a given time slot.

Consumers may have relatively large jobs compared to the bundle size. To avoid unfavorable constellations and as a first step towards workflow representation, we allow consumers to request multiple sub jobs, henceforth called "job parts"[7]. The job specification is extended by including the parameter α^{nj}, which represents the number of job parts that should be allocated for job satisfaction as well as the *parallelization* parameter to express the simultaneous allocation of job parts. In spite of job allocation to multiple bundles, each identical part is assigned to exactly one bundle at a time. This *coupling* enables the avoidance of data transfers between multiple sites (Figure 1).

In general, let $R = \{1, \ldots, \overline{r}\}$ be the set of resources and $A_r = \{1, \ldots, \overline{a_r}\}$ the set of resource attributes with the maximal quantities $\overline{r}$ and $\overline{a_r}$. To distinguish between the demand and the supply side, A_r applies to resource attributes of jobs and $D_r = \{1, \ldots, \overline{d_r}\}$ to bundles. Regarding our scenario, two resources $R = \{1, 2\}$ are traded containing three attributes per resource ($A_r = \{1, 2, 3\}$). We consider two resources: computation and storage service. Attribute a_{11}^{nj} indicates the required quantity of CPUs, a_{12}^{nj} the CPU speed (in GHz) and the RAM size (in MB). Storage service (resource 2) contains the attribute a_{21}^{nj} that represents

[7]Note that this can be considered as a restricted form of the AND operator.

Table 1. Consumer bids.

n	j^n	v^{nj}	α^{nj}	e^{nj}	l^{nj}	q^{nj}	γ^{nj}	a_{11}^{nj}	a_{12}^{nj}	a_{13}^{nj}	a_{21}^{nj}	a_{22}^{nj}	a_{23}^{nj}
1	1	60	2	1	4	3	1	2	2	2048	1	3	100
	2	10	1	3	5	1	0	2	2	3096	2	1	50
2	1	70	1	1	4	3	0	4	1	512	5	1	80
	2	40	2	2	4	3	1	1	2	2048	1	3	80
3	1	90	3	1	5	4	0	2	2	2048	2	2	50

Table 2. Supplier bids.

m	b^m	p^{mb}	c^{mb}	f^{mb}	d_{11}^{nj}	d_{12}^{nj}	d_{13}^{nj}	d_{21}^{nj}	d_{22}^{nj}	d_{23}^{nj}
1	1	3	1	5	5	2	3096	6	3	80
	2	4	1	5	5	2	3096	2	3	80
2	1	6	2	5	3	3	3096	7	2	100
	2	5	1	3	6	1	1024	8	3	80

the number of hard disks, a_{22}^{nj} that indicates the throughput (in MB/s) and a_{23}^{nj} that gives the size of the disks.

The concept of discrete time slots of fixed and uniform length within an allocation horizon $T = \{1 \ldots, \bar{t}\}$ is implemented in which agents indicate availability time frames and consumers additionally the duration of jobs. For successful allocation, this duration q^{nj} (which is bounded by the execution time frame $[e^{nj}, l^{nj}]$) has to meet the bundle's availability time range $[c^{mb}, f^{mb}]$.

Job preferences are indicated as valuations v^{nj} and reservation prices for bundles per time slot with p^{mb}. This definition, however, in combination with bundle indivisibility within a single time slot provokes overpayment in the case that consumers ask fractions of bundles. Parameter γ^{nj} takes the value 1 if parallelization of all job parts within a job is desired and $\gamma^{nj} = 0$ otherwise.

In conclusion, the elements of the bid expression are connected in the following way:

Job $= \{(\{a_{11}^{nj}, a_{12}^{nj}, a_{13}^{nj}\}, \{a_{21}^{nj}, a_{22}^{nj}, a_{23}^{nj}\}, e^{nj}, l^{nj}, q^{nj}), \alpha^{nj}, v^{nj}, \gamma^{nj}\}$

Bundle $= \{(\{d_{11}^{mb}, d_{12}^{mb}, d_{13}^{mb}\}, \{d_{21}^{mb}, d_{22}^{mb}, d_{23}^{mb}\}, c^{mb}, f^{mb}), p^{mb}\}$

Sample bids are demonstrated by considering 5 time slots, 3 consumers and 2 suppliers. Table 1 gives the consumer bids with associated jobs, job parts, time and allocation characteristics.

Correspondingly, Table 2 gives the specification of supplier bids.

4.2 The allocation mechanism

The resulting allocation problem represents a mixed-integer optimization problem.

The objective function 4.1 aims at maximizing the social welfare of participants. This is represented by the difference of the sum of job valuations v^{nj} for allocated jobs and the sum of reservation prices p^{mb} of allocated bundles in the associated time slots. Less formally, our aim is to allocate the most valuable job requests to the cheapest resources that satisfy the technical job requirements.

$$\max V = \sum_{n\in N}\sum_{j\in J^n} z^{nj} v^{nj} - \sum_{m\in M}\sum_{b\in B^m}\sum_{n\in N}\sum_{j\in J^n}\sum_{\alpha\in Q^{nj}}\sum_{t\in T} y^{nj\alpha}_{mb,t} p^{mb} q^{nj} \tag{4.1}$$

$$\text{s. t.} \quad z^{nj} \in \{0,1\}\,, \quad \forall n\in N\,, \quad \forall j\in J^n \tag{4.2}$$

$$y^{nj\alpha}_{mb,t} \in \{0,1\}\,, \quad \forall n\in N\,, \quad \forall j\in J^n\,, \quad \forall \alpha\in Q^{nj}\,, \quad \forall m\in M\,, \quad \forall b\in B^m\,, \quad \forall t\in T \tag{4.3}$$

$$\sum_{j\in J^n} z^{nj} \le 1\,, \quad \forall n \in N \tag{4.4}$$

$$\sum_{m\in M}\sum_{b\in B^m}\sum_{t\in T} y^{nj\alpha}_{mb,t} - z^{nj} = 0\,, \quad \forall n\in N\,, \quad \forall j\in J^n\,, \quad \forall \alpha\in Q^{nj} \tag{4.5}$$

$$\sum_{n\in N}\sum_{j\in J^n}\sum_{\alpha\in Q^{nj}}\sum_{t'=\max\{1,t-q^{nj}+1\}}^{t} y^{nj\alpha}_{mb,t'} \le 1\,, \quad \forall m\in M\,, \quad \forall b\in B^m\,, \quad \forall t\in[c^{mb}, f^{mb}] \tag{4.6}$$

$$\gamma^{nj}\sum_{m\in M}\sum_{b\in B^m} y^{nj\alpha}_{mb,t} - \gamma^{nj}\sum_{m\in M}\sum_{b\in B^m} y^{nj\alpha'}_{mb,t} = 0\,, \quad \forall n\in N\,, \quad \forall j\in J^n\,, \quad \forall t\in[e^{nj}, l^{nj}]\,, \quad \forall \alpha,\alpha'\in Q^{nj}, \alpha\neq\alpha\prime \tag{4.7}$$

$$y^{nj\alpha}_{mb,t} = 0\,, \quad \forall m\in M\,, \quad \forall b\in B^m\,, \quad \forall n\in N\,, \quad \forall j\in J^n\,, \quad \forall \alpha\in Q^{nj} \quad \forall t \notin \left[\max(e^{nj}, c^{mb}), \min\{l^{nj}, f^{mb}\} - q^{nj} + 1\right] \tag{4.8}$$

$$y^{nj\alpha}_{mb,t} = 0\,, \quad \forall m\in M\,, \quad \forall b\in B^m\,, \quad \forall n\in N\,, \quad \forall j\in J^n\,, \quad \forall \alpha\in Q^{nj} \quad \forall t\in T\,, \quad \forall r\in R\,, \quad \forall i\in A_r \setminus \{k\in A_r \mid a^{nj}_{rk} > d^{mb}_{rk}\}\,. \tag{4.9}$$

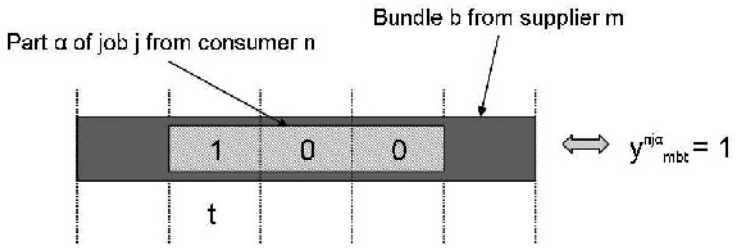

FIGURE 2. Usage of variable y.

First, two decision variables are introduced (Constraints 4.2 and 4.3). The binary variable z^{nj} specifies if job $j \in J^n$ of consumer bid n is allocated ($z^{nj} = 1$) or not ($z^{nj} = 0$).The second binary variable $y^{nj\alpha}_{mb,t}$ indicates whether job part $\alpha \in Q^{nj}$ of job j of consumer bid n is allocated to bundle $b \in B^m$ of supplier bid b in time slot t ($y^{nj\alpha}_{mb,t} = 1$) or not ($y^{nj\alpha}_{mb,t} = 0$).

Constraints 4.4 ensure the representation substitute in job allocation (XOR). At most one job j of consumer bid n can be allocated.

Constraints 4.5 force the variables j and z to be coherent. Any job part α of an allocated consumer bid ($z^{nj} = 1$) is assigned only once over all bundles and time slots. This guarantees coupling.

Consistency in time and avoidance of overlapping jobs are expressed by Constraints 4.6. For each supplier and bundle, only time slots within the availability time frame $t \in [c^{mb}, f^{mb}]$ of the bundle are considered. For each job part α, only one time slot of the allocation is marked by setting $y^{nj\alpha}_{mb,t} = 1$. This applies only to this single time slot across all job parts (cf. Figure 2). Due to the check of time slot $t' = t - q^{nj} + 1$, in the following $q^{nj} - 1$ time slots no other job part can be allocated and the overlapping of jobs on a single bundle is avoided. At the same time, the allocation of $q^{nj} - 1$ time slots in succession ensures consistency in time.

Constraints 4.7 guarantee the allocation of job parts of a given job in exactly the same time slots within the job's time frame, i.e. in parallel. Indeed, these constraints are a tautology when $\gamma^{nj} = 0$ and activate parallelization in the case of $\gamma^{nj} = 1$.

Time constraints 4.8 ensure allocations within the period, in which job and bundle time frames overlap. As $y^{nj\alpha}_{mb,t} = 1$ indicates that job part α starts in time slot t, the constraints force variable y to be zero if time intervals do not match.

Constraints 4.9 consider the quality and quantity attributes. Allocation of job part α to bundle b is impossible if the value of the job's attribute i (a^{nj}_{ri}) is greater than the value of the respective attribute of bundle b (d^{mb}_{ri}).

The schedule of a sample allocation is given in Table 3. Job 1 of consumer bid 3, which consists of 3 job parts, is allocated to bundles 1 and 2 of supplier bid 1 in time slots 1 to 4 and to bundle 1 of supplier bid 2 in time slots 2 to 5. Job 2 of consumer 1 is allocated to bundle 1 of supplier 1 in time slot 5. Consequently,

Table 3. Schedule of sample allocation.

Supplier			Consumer		Time slot t	Partial welfare
m	b^m	p^{mb}	$n/j^n/\alpha^{nj}$	v^{nj} per α^{nj}		
1	1	3	3/1/1	30	1-4	18
			1/2/1	10	5	7
	2	4	3/1/2	30	1-4	14
2	1	6	3/1/3	30	2-5	6
	2	5	2/1/1	70	1-3	55
					Social welfare	100

bundle 1 of supplier 1 is reserved in its complete availability time in contrast to bundle 2 which is idle in time slot 5. As there are no more unsatisfied consumers, this time slot stays idle.

4.3 The pricing

This section addresses the question of how to calculate payments for winning requests and offers after the end of the auction. The well-known Myerson-Satterthwaite impossibility theorem demonstrates that, in a very general setting, no exchange can be allocation-efficient, budget-balanced, individually rational and (Bayes-Nash) incentive compatible at the same time [9]. Thus, the mechanism designer has to prioritize and trade-off economic properties depending on the desired outcome.

Generally, pricing schemes are based on the distribution of overall welfare. The prominent Vickrey-Clarke-Groves (VCG) mechanism is the only mechanism combining incentive compatibility, allocative efficiency and individual rationality [13]. The idea is that each agent i obtains a Vickrey discount corresponding to the gain in welfare which results from his participation. The payment calculation of the VCG pricing scheme, however, does not balance the budget in bilateral trades and will thus generally need to be subsidized by outside payments.

If the WDP is solved to optimum, the VCG pricing rule yields incentive compatible payments and motivates agents to report their true valuations [4]. This leads to efficient allocations. Requiring long runtimes by having to solve the WDP $n-1$ times, the VCG pricing rule may be used for batch mechanisms but not for interactive applications that demand fast market clearings. Due to the permanently needed subsidization from the outside, other pricing mechanisms such as *approximate VCG pricing and K-pricing* have been developed meeting the requirement of budget balance. The approach of [10] is based on the VCG mechanism and adds payment rules that minimize the distance to Vickrey discounts for some metrics. Approximate VCG pricing meets the economic properties of individual rationality and budget balance. The mechanism, however, is not incentive compatible anymore. Due to the complex calculation of payments, VCG-pricing and

TABLE 4. Payment structure under different pricing schemes.

Pricing scheme	VCG	Approximate VCG	K-Pricing		
			$k = 0.4$	$k = 0.5$	$k = 0.6$
n_1	-3	-10	-7.2	-6.5	-5.8
n_2	-15	-30	-48	-42.5	-37
n_3	-58	-73	-74.8	-71	-67.2
m_1	72	57	$+50.4$	$+47.2$	$+44$
m_2	71	56	$+79.6$	$+72.8$	$+66$
Budget	-67	0	0	0	0

approximate VCG pricing are not applicable in practice. Thus, K-Pricing was introduced in [13]. The basic idea is to distribute the welfare generated by the allocation algorithm between resource requesters and utility computing providers according to a factor $k \in [0, 1]$. For instance, assume an allocation of resources from a specific provider to a specific requester. The buyer values these resources at \$10 while the provider has a reservation price of \$5. Then the (local) welfare is $\$10 - \$5 = \$5$ and $k \cdot \$5$ of the surplus is allotted to the requester – thus having to pay $\$10 - k \cdot \$5$ – and $(1 - k) \cdot \$5$ is allotted to the provider – thus receiving $\$5 + (1 - k) \cdot \$5$. Besides allowing fairness considerations, the main advantage of K-Pricing is that it can be determined in polynomial runtime. On the downside, however, it only yields approximately truthful prices and payments on both sides of the market.

Table 4 exemplifies the payment structure under VCG pricing, approximate VCG pricing and K-pricing of the sample allocation schedule of Subsection 4.2.

The economic property of budget balance is outstanding. While VCG pricing yields a negative budget of -67, K-pricing and approximate VCG pricing achieve a balanced budget. As in the scenario subsidization from the outside is impossible, VCG pricing is not applicable.

5. Evaluation

The outcome of the mechanism is an efficient allocation of computation and storage services. In this section, the mechanism's performance is examined with respect to the influences of the problem size and parameter choices. The numerical experiment is processed with the standard solver for linear optimization problems ILOG CPLEX 10.0 on a machine with a 3 GHz processor and 1.5 GB RAM. The tests aim at identifying specific impacts of parameters so that conclusions can be drawn for the configuration of markets.

Table 5. Distribution and ranges of parameters.

Parameter	Distribution and Ranges
Agents	fixed (70 suppliers and 60 consumers)
Bundles	$U(1,3)$
Jobs, Job parts	$U(1,5), U(1,6)$
Time Slots	8 or 12
Reservation price, Valuation	$N(7,1), U(7,12)$
Parallelization	Binomial with $p = 0.3$

5.1 Data generation

As the evaluation cannot bear on reported user data, the input data is based on artificial problem instances. For such test cases with unknown data characteristics the uniform distribution U is widely used. Table 5 shows the distribution and ranges of parameters.

The evaluation procedure starts with the generation of bid streams, so-called *settings*, each comprising a set of 30 randomly generated test instances as specified in Table 5. The mean value is calculated over the test instances of a setting to soften random outcomes and to avoid the influence of outliers.

5.2 Analysis

The complexity of the WDP is the decisive factor for the model's performance on the one hand and its limitations of computational tractability on the other hand. The simplest case of the WDP can be seen as a reduction to the one-dimensional *Knapsack Problem* that is known to be NP-hard.

Besides its theoretical complexity, the practical complexity depends on the choice of parameter values and supposes characteristic behaviours concerning e.g. runtime. The required runtime does not only depend on the size of the instance with regard to the number of bids, bundles, jobs, job parts and time slots but also on the specific composition of time characteristics, resource attributes and allocation parameters [12].

The characteristic of NP-hard problems is the computational intractability within a meaningful time frame even for small instances. Regarding the scenario, it is acceptable to approximate the welfare maximizing solution and to achieve an allocation within a "reasonable" time frame. Two stop conditions are defined for the test: the achievement of a threshold value of 5% that represents the proximity to the projected optimal welfare and a time limit of 20 minutes. The solution of each test is measured by two performance indicators μ and λ. Thereby, μ is the average runtime over all instances to attain a value less than the threshold to the optimal welfare within at most 20 minutes. λ denotes the ratio of the number of "hard" instances to the total number of instances in a setting. An instance is

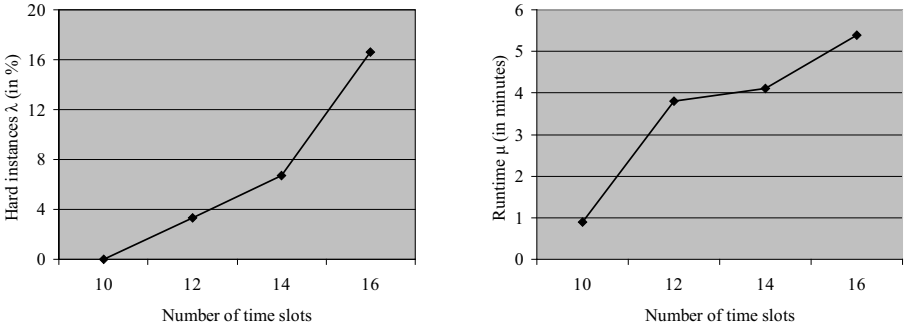

FIGURE 3. Development of runtime μ and hard instances λ with an increasing number of time slots.

defined as hard when it does not achieve a value of less than the threshold value within a runtime of 20 minutes. The runtime of hard instances is included in the duration of 20 minutes.

Test 1 – Increasing number of time slots

The objective of this test is to provide a statement on the mechanism's scalability with an increasing number of time slots. Thereby, the impact of uniformly extended length of each time slot is considered within the allocation horizon of constant length. The number of time slots increases across the four settings, starting from $\bar{t} = 10$. The settings differ in the number of time slots and consequently in parameters that depend on these numbers, such as the time parameters $e^{nj}, l^{nj}, q^{nj}, c^{mb}, f^{mb}$, valuations v^{nj}, and reservation prices p^{mb}.

The rising curves (Figure 3) of the runtime performance indicators μ and λ indicate the increasing complexity of problem solving with an increasing number of time slots. In case of 10 time slots, there is a 100% probability that the mechanism yields the optimal solution with a threshold value less than 5% within 20 minutes. This ratio drops slowly to 93% for 14 time slots and shows then a sharp reduction to 83% for 16 time slots. Analogously, the runtime increases from less than one minute to 5.4 minutes on average for 16 time slots.

As all other dimension parameters are kept constant and quality parameters are adjusted proportionally, there are no further influences. Thus the test shows that an increase of time slots leads to rising complexity and runtime. Consequently, the trade-off between increasing runtime and flexibility in expression of time attributes has to be considered to configure the time concept of the market.

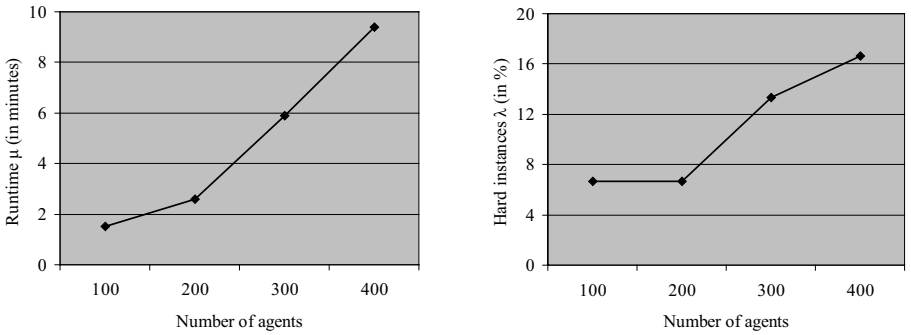

FIGURE 4. Development of runtime μ and hard instances λ with an increasing number of agents. The ratio of consumers to suppliers keeps constantly 2:3.

Test 2 – Increasing number of agents

In this test, the number of agents is raised in steps of 100 starting from $\overline{n}+\overline{m} = 100$ through to 400 by keeping the ratio of consumers to suppliers of 2:3 constant. As the number of jobs $\overline{j} = 5$ and bundles $\overline{b} = 3$ is fixed, the ratio of supply to demand increases proportionally.

In case of 40 consumers and 60 suppliers, the algorithm needs 1.5 minutes on average to fall below the threshold and to yield the best solution found. With an increasing number of agents, the curve of rises slowly until 200 agents are reached and soars in the range of 200 to 400 agents. For 400 agents the runtime μ goes up to 9.4 minutes (Figure 4) and the curve of λ increases.

Approaching 500 agents, the problem size exceeds the limit of computational tractability. The need for a solution that remains within a reasonable time frame, implies the need for an upper bound to limit the number of agents. As the curves of runtime and hard instances increase drastically between 200 and 300 agents, a number of about 250 agents could be an optimal bound for which the problem is computationally tractable.

Test 3 – Influence of the quality parameters parallelization and job parts

Starting with the impact of parallelization, the two extreme cases are examined: activated parallelization in all jobs ($\gamma^{nj} = 1$) that reduces drastically the feasible set of allocations and the same setting without parallelization restrictions ($\gamma^{nj} = 0$).

The determining factors are Constraints 4.7 that only generate restrictions in case of activated parallelization. The activated parallelization in 100% of the jobs nearly avoids achieving a non-hard instance in the setting; 87% are hard instances. Hence the runtime μ is long too and takes 19 minutes (Figure 5 on

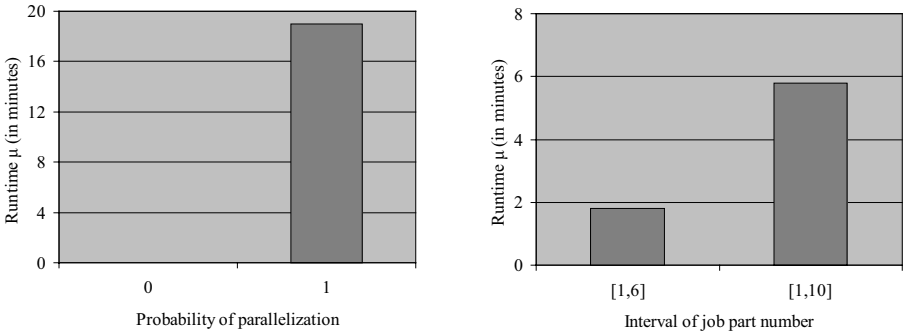

FIGURE 5. Influence of parallelization restrictions and an increasing number of job parts on runtime μ.

the left). On the contrary, there are no hard instances in the setting without parallelization restrictions and the runtime μ is negligible with a value close to zero (0.03 minutes).

In conclusion, the simultaneous allocation of job parts is out of all proportion to complexity. The increased user utility of activated parallelization has to be opposed to its downgrading effect on scalability.

Secondly, the influence of an increasing number of job parts is considered. Starting with a number of job part of the interval $[1, 6]$, the maximum quantity is increased to $\overline{\alpha} = 10$.The determining factor that affects the problem complexity are again Constraints 4.7. They generate a corresponding α' to each α in case of activated parallelization. Additionally, most of the other constraints run through $\overline{\alpha}$ times. Thus the more job parts exist, the more restrictions are generated. The number of hard instances increases from 0% for interval $[1, 6]$ up to 13% for interval $[1, 10]$ and at the same time, the runtime triples from 1.8 to 5.8 minutes (Figure 5 on the right). As the possibility to indicate numerous job parts is a crucial factor of the scenario, market configuration has to provide a sufficiently large maximum number of job parts.

6. Summary and future work

At the core of this paper, we elaborated a model for economic resource allocation that supports the trading of complex services and the specification of workflows. The model can be configured to implement scenario-dependent requirements such as coupling and parallelization. The model was evaluated with respect to the market configuration. To this end, three numerical tests were carried out to investigate different influences of dimension and quality parameters on the average runtime and problem complexity. Although the model scaled well with an increasing number of time slots and agents, the scope of the problem exceeded the limit of computational tractability when 500 agents were approached. Quality parameters

added enormous complexity to the problem. In particular, the combination of a large number of job parts and the parallelization restrictions reduce the model's performance.

Our work suggests two main areas for future research, the *allocation mechanism* and the *pricing rules*. As the evaluation of the mechanism could not rely on reported input data, it would be of interest to run tests with real workload data. Moreover, a next step in the model evaluation could be a numerical comparison with MACE. As MACE is formulated as a mixed integer program and the proposed model in this work as an integer program, a benchmark test can provide information about the influence of the problem formulation on the mechanism's performance. Finally, future extensions to the allocation mechanism could consider the involvement of network distances and capacities as well as the representation of complex workflows. As regards the pricing, the next step will be the design of a pricing scheme which is specifically tailored towards our application scenario. An interesting approach may be the implementation of an iterative process for price determination. Iterative combinatorial auctions provide this functionality by incrementally approximating equilibria based on supply and demand.

References

[1] A. Au Young, B. Chun, A. Snoeren, A. Vahdat: *Resource Allocation in Federated Distributed Computing Infrastructures*, Proceedings of the 1st Workshop on Operating Systems and Architectural Support for the On-demand IT Infrastructure. (2004).

[2] R. Bapna, S. Das, R. Garfinkel, J. Stallaert: *A Market Design for Grid Computing*, Journal on Computing, Forthcoming available at SSRN: http://ssrn.com/abstract=927036. (2005).

[3] B. Chun, D. Culler: *Market-based Proportional Resource Sharing for Cluster*, Millenium Project Research Report. (1999).

[4] S. De Vries, R. Vohra: *Combinatorial Auctions: A Survey, Journal on Computing*, **15** (3) (2003), 284–309.

[5] W. Fellows, S. Wallage: *Grid Computing* – The state of the market, 451 GARS – Report 14 Executive Overview. (2007).

[6] I. Foster, C. Kesselmann, S. Tuecke, *The Anatomy of the Grid: Enabling Scalable Virtual Organizations* The International Journal of High Performance Computing Applications **15** (3) (2001), 200–222.

[7] J. Gomoluch, M. Schroeder: *Market-based Resource Allocation for Grid Computing: A Model and Simulation*, Proceedings of the 1st International Workshop on Middleware for Grid Computing, Rio de Janeiro, Brazil.

[8] K. Lai, L. Rasmusson, E. Adar, L. Zhang, B. A. Huberman: *Tycoon: An implementation of a distributed, market-based resource allocation system, Multiagent and Grid Systems*, **1** (3) (2005), 168–182.

[9] R. Myerson, M. Satterthwaite: *Efficient Mechanisms for Bilateral Trading*, Journal of Economic Theory, **29** (2) (2005), 265–281.

[10] D. Parkes, J. Kalagnanam, M. Eso: Achieving Budget-Balance with Vickrey-Based Payment Schemes in Combinatorial Exchanges, IBM Research Report RC 22218 W0110-065. (2001).

[11] O. Regev, N. Nisan: *The POPCORN Market – An Online Market for Computational Resources*, (2000), 148–157.

[12] T. Sandholm: *Optimal Winner Determination Algorithms*, P. Cramton, Y. Shoham, R. Steinberg: *Combinatorial Auctions*, MIT Press. (2006).

[13] B. Schnizler, D. Neumann, D. Veit, C. Weinhardt: *Trading Grid Services – A Multi-attribute Combinatorial Approach*, European Journal of Operational Research, forthcoming (2006).

[14] C. Waldspurger, T. Hogg, B. Huberman, J. Kephart, W. Stornetta: Spawn: *A Distributed Computational Economy, IEEE Transactions on Software Engineering*, **18** (2) (1992), 103–117.

Melanie Moßmann
Goethestr.18
67251 Freinsheim
Germany
e-mail: melanie.mossmann@gmx.de

Jochen Stößer
Universität Karlsruhe (TH)
Englerstr. 14
76131 Karlsruhe
Germany
e-mail: stoesser@iism.uni-karlsruhe.de

Adam Ouorou
Orange Labs
38 Rue du General Leclerc
92130 Issy les Moulineaux
France
e-mail: adam.ouorou@orange-ftgroup.com

Eric Gourdin
Orange Labs
38 Rue du General Leclerc
92130 Issy les Moulineaux
France
e-mail: eric.gourdin@orange-ftgroup.com

Ruby Krishnaswamy
Orange Labs
38 Rue du General Leclerc
92130 Issy les Moulineaux
France
e-mail: ruby.krishnaswamy@orange-ftgroup.com

Dirk Neumann
Chair for Information Systems Research
Albert-Ludwigs-Universität Freiburg
Kollegiengebäude II
Platz der Alten Synagoge
79085 Freiburg
Germany
e-mail: dirk.neumann@is.uni-freiburg.de

Economic Models and Algorithms for Distributed Systems, 239–256
Book Series: Autonomic Systems

Heuristic Scheduling in Grid Environments: Reducing the Operational Energy Demand

Christian Bodenstein

Abstract. In a world where more and more businesses seem to trade in an online market, the supply of online services to the ever-growing demand could quickly reach its capacity limits. Online service providers may find themselves maxed out at peak operation levels during high-traffic timeslots but too little demand during low-traffic timeslots, although the latter is becoming less frequent. At this point deciding which user is allocated what level of service becomes essential. The concept of Grid computing could offer a meaningful alternative to conventional super-computing centres. Not only can Grids reach the same computing speeds as some of the fastest supercomputers, but distributed computing harbors a great energy-saving potential. When scheduling projects in such a Grid environment however, simply assigning one process to a system becomes so complex in calculation that schedules are often too late to execute, rendering their optimizations useless. Current schedulers attempt to maximize the utility, given some sort of constraint, often reverting to heuristics. This optimization often comes at the cost of environmental impact, in this case CO_2 emissions. This work proposes an alternate model of energy efficient scheduling while keeping a respectable amount of economic incentives untouched. Using this model, it is possible to reduce the total energy consumed by a Grid environment using 'just-in-time' flowtime management, paired with ranking nodes by efficiency.

Mathematics Subject Classification (2000). Primary 68Q10; secondery 68Q85.

Keywords. Combinatorial exchange, complex grid services, simulation

1. Introduction

The world is rapidly growing into an integrated network of computers sharing information [12]. In order to keep up with sustained business growth, companies and

organizations constantly need to improve their information technology infrastructure. These improvements often entail upgrading current systems to be better, faster, and larger. In most cases however, these resources are not required permanently, leaving systems often running idle. As a result tasks are often outsourced to third parties, and their services are purchased when the demand is high; the resources are rented. This omits the problem of under-utilization, but brings up problems in administration, leaving most companies with few other options but to buy expensive systems, and leave them running idle for most of the time.

For this reason, computing Grids have become increasingly appealing to organizations looking for high computing power for only short periods of time – not only because they do not have the capacity to invest in their own computing infrastructure, but also because they do not have a constant demand for the computing power [3]. Grid computing integrates these idle computing resources across administrative boundaries, comprised of a set of computers linked by a network. A possible solution to this dilemma has been to deliver computing resources as a service or utility wherever needed. Where previously systems could be rented, companies now offer computing utility "as a service".

A standard Desktop PC, like the one used to write this work, uses between 50 (in standby) and 300 Watts of energy. By adding the power required to produce PC's, to lay communication cables, power server clusters and wireless networks, and all other factors required, today, the information and communications sector in Germany consumes more than eight percent of total electricity consumption in the country alone. In the USA, the officially known computer centers alone already devour 45 Billion KWh – 1.2 percent of national power consumption. This energy consumption, in itself already a major cost driving factor, further burdens the ecology. Gartner estimates that combined CO_2 emissions of the global ICT industry from the manufacture, distribution and use of IT equipment generates about 2% of global emissions. In other words, it has reached the "CO2-footprint" of the aviation industry. To put this seemingly small percentage into perspective, volcanoes and other natural phenomenon produce only about 1% [20].

In other carbon-industries, pigou-taxes have been used to regulate CO_2 emission [2, 9, 33]. This however becomes increasingly difficult in the ICT-industry, as it is only indirectly responsible for the energy-based CO_2 emission volume. Although this might be an option, undoubtedly a direct energy-saving mechanism is more desired: Experts look towards green computing solutions to minimize the energy consumed by the ICT industry. If we wish to combat the threat of global warming and sustain our way of life, every possibility at reducing the CO_2 emissions must be pursued, regardless of its economic, or taxing implications. These goals are inherently co-aligned. Not only will consuming less electricity lessen the CO_2 emissions, a moral duty, but also by reducing power consumption, companies harbor great potential of reducing production costs, ultimately saving a lot of money. The research question posed by this work is thus: 'Can existing scheduling

models be altered in such a way that they allow for more energy efficiency in the Grid environments, without loss of incentive compatibility and revenue?'

This question will be answered in this work, as it is novel, in that:

- It proposes the use of a green heuristic allocation mechanism which exhibits desirable strategic properties.
- It points to possible limitations of the mechanism and suggests strategies that may mitigate these drawbacks.

This work is structured as follows. After a brief introduction to the problem at hand, Section 2 will outlay some related work in energy efficient Grid scheduling. Section 3 shows a model which aims to; minimize the energy consumed by a Grid application, while still keeping an acceptable level of economic welfare. Following the formal analysis, a heuristic allocation scheme is presented. Section 4 discusses possible incentives and analyses possible solution strategies to counter them. Section 5 concludes this work and outlooks questions raised during model conception.

2. Related work

A foundation of Grid technologies and a good introduction to Grids was given by [13] who define the Grid as the concept of "coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations". Since then not only has the idea of the Grid evolved, but it has also often been misunderstood, labeling near everything between networking to Artificial Intelligence as "the Grid". Recently, a consensus arose, narrowing the concept of Grid computing down to a well-defined technology base that addresses a problem at hand and offers a solution. The objective of standard technical schedulers is usually to maximize resource utilization and often to balance the system load. Few schedulers however include energy efficiency in the goals.

Work on energy-aware scheduling although scarce, was first discussed by [35], assuming that $power = speed^{\beta}$ for some $\beta \leq 1$. Based on this assumption, less energy is consumed over the duration of a process if the speed of the processors is reduced, as shown in recent work by [1]. Research has also gone into hardware inefficiencies and heat build-up in computing clusters. Research on this end has determined that most energy flowing into systems is given off as heat, that only about 10% (percentage varies in hardware complexity) of energy is actually available for energy optimization [25]. The above work focused on the technical aspect of scheduling in Grids. To be economically viable, Grids needed a pricing scheme. Similar modeling simulations have been experimented by [11].

This work is unique, in that previous attempts to incorporate energy conservation and flow-time management, which remain orthogonal objectives, have always involved either setting the amount of energy available constant or minimizing time-based objective functions, while in this work the constant factor has

been the project deadline. This has the advantage that, while energy availability procedures allow the energy level to be specifically set, the scheduler can definitely assign jobs which have to be executed.

These attempts include the goal of minimizing the average percentage of energy consumed by the system, while meeting an execution time constraint similar to [27] where the goal of the allocation is to minimize the average percentage of energy consumed by the application to execute across the machines in the ad-hoc Grid, while meeting an application execution time constraint. [24] presented an algorithm maximizing the rewards under a limited energy budget, without exceeding the deadlines or available energy. These algorithms were especially used to schedule on portable devices, where battery operation is limited.

When it comes to pricing goods with incomplete information, auctions have often been a part of the solution in some way or another. A foundation was laid by [3] and extended by several authors, Grid market models ranging from commodity exchanges [36] through combinatorial double auctions [26] to heuristics [29]. Mostly, CPU power from the nodes have been auctioned to job requesters, with the agents requesting jobs, paying the node suppliers through a scheduling agent which allocates the job to a specific node. Likewise, auctions have played a major role in the integration of market mechanisms into Grid systems, effectively allowing for cooperative and competitive trading of resources such as CPU cycles [6]. Examples of these are the Popcorn market for computing power [23], the proportional market-based share approach [7] which all trade computing power. [34] first proposed trading off computing time using a Vickrey auction.

This work is largely founded on the work by [29]. In an attempt to include energy costs in their model, further inspired by [1] the model was successfully adapted and further generalized.

3. The model

This section elaborates on a market-based model which consists of agents submitting time-sensitive computing jobs to the Grid, nodes supplying computing resources, specifying the times that their machines are available, and an automated Grid operator deciding which job to allocate to which node. We will first introduce the bidding language, which acts as a communication medium to indicate participant's willingness to trade, followed by a formal mathematical representation of the allocation problem. Secondly, we will elaborate on a heuristic allocation algorithm which achieves a near perfect clone of its NP-hard counterpart.

3.1 The setting

In this setting computing power is the central scarce resource to be traded in an offline Grid environment. There are two participating parties in the market: requesters, who wish to obtain computing resources, thereby maximizing their

private utility functions, and providers who supply the market with resources, maximizing their profits. The market objective is to reduce the energy costs over all trades. The market acts as a bulletin board for participants, meaning it collects offers and requests for a short period of time, before allocating the resources and clearing the board and once again collecting further posts. These periods will be referred to as *phases* (See also [1]) The allocation is primarily handled by a sealed bid auction mechanism, where requesters and suppliers do not know the other users' requests and offers. (See also [19, 29, 32, 34])

For the resource offers, let N describe the set of resource offers ("node") made by the providers which the mechanism has collected over a phase. When submitting such a resource offer $n \in \mathbb{N}$ the providers post the bid $(r_n, c_n, f_n, l_n, \epsilon_n)$ to the market board. $r_n \in \mathbb{R}_0^+$ specifies the node reservation price per computing power per timeslot, depicted in arbitrary *monetary units* (MU). $c_n \in \mathbb{N}$ and $m_n \in \mathbb{N}$ are the computing units (CPU) and memory available per time slot, where $f_n \in \mathbb{N}$ is the first available time slot in which these resources are available and $l_n \in \mathbb{N}$ is the last time slot; the time where the resources can be accessed [29]. Additionally the providers submit their energy efficiency level, in the form of a power consumption bid $\epsilon_n \in \mathbb{R}_0^+$. Each node can be seen as a perfectly divisible virtual machine capable of executing multiple jobs in parallel as proposed by [4]. Providers are assumed to have complete information about their availability. When scheduling therefore, the mechanism can compute with perfect foresight an optimal allocation scheme.

For the "agents", let J represent requests to execute a job $j \in J$. The agents report their tuple $(b_j, C_j, s_j, d_j) \in \mathbb{N}$ to the system. b_j is the agents bid per computing resources per unit time, depicted in *monetary units* (MU). By assumption, the agents submit their bid based on the knowledge that their jobs will be delivered 'just-in-time'. Agents are therefore indifferent between jobs delivered earlier, or on their designated deadline. Hence, the bid submitted by an agent reflects his valuation based only on the job itself since $\frac{\partial b_j}{\partial (d_j - s_j)} = 0$. C_j is the total processing units required in CPU's or CPU cycles and is therefore the only variable which b_j is dependent on. s_j is the start time and the job deadline is given by d_j. Memory requirements are omitted in this model due to possible scaling issues between processing speeds and memory usage at increased speeds. It is therefore assumed that all nodes have sufficient memory to store multiple tasks. Each job is submitted and can only be executed in its entirety, allowing only complete allocation to the resources available in the designated start and end time, or none at all.

In order to simulate this scenario, the resource requirements and characteristics were all drawn randomly using normal distributions, similar to the data generation recommended by [11] for the creation of parameters to simulate the model. The parameter values are always positive, since technically, there is no such thing as a negative runtime and the distribution is assumed to be skewed, implying, that there are more small values and fewer large values. To achieve this simulation in the model, a lognormal distribution $log\mathbb{N}(\mu, \sigma^2)$ is used. The initial settings are drawn from Table 1.

TABLE 1. Simulation settings.

Parameter	Resource Requesters	Resource Providers
Computing Power	Lognormal(25, .25)	Lognormal(65, .25)
Start time	Lognormal(2, .06)	Lognormal(2, .06)
End time	Lognormal(6, .06)	Lognormal(8, .06)
Valuation	Uniform (15, 10)	Uniform (10, 6)
Energy costs	N/A	Uniform (0, 10)

TABLE 2. Sample job requests and node offers.

Node	r_n	c_n	f_n	l_n	ϵ_n	**Job**	b_j	C_j	s_j	d_j
N1	11	35	1	8	9	**J1**	20	36	3	5
N2	12	44	2	8	4	**J2**	17	42	5	7
N3	13	22	1	8	12	**J3**	11	172	2	5
						J4	13	24	1	8
						J5	24	84	1	6
						J6	12	90	4	8

Example 4. Table 2 lists an example of requests and offers which have been submitted. Node N1 requires at least 11 MU (his reservation price) per unit of computing power and offers a total of 35 units of computing power throughout time slots 1 through 8. Job J1 on the other hand is willing to pay 20 MU per computing power per time slot, and is requested during time slot 3 to 5.

Current schedulers would now aim to maximize the resource allocation, or to balance the overall system load. In this model the scheduler will additionally be the instrument by which the energy consumption of the Grid is decided. By being able to control both the processing speeds (implicitly), and the allocation of jobs to nodes, the largest potential in energy conservation lies within the scheduling mechanism.

Proposition 3.1. *Given the exponential relation between processing speeds and power consumption, it is always better to spread out a job over multiple periods, running the processor at a reduced speed, than to run the same job at full power in a single period.*

Proof. Given the relation set by [1] $P(s) = s^a$, assume, for $a \geq 2$, a job which requires k CPU's can be either processed in a single period, running at full capacity of s CPU on a node, or in 2 periods, running at half capacity. $P(s) = s^2; P(s/2) = (s^2/4)$ hence $P(s) \leq P(s/2)$. □

Since the most effective means to conserve energy is to not use the Grid, solving the problem purely by minimizing energy consumption is not a feasible solution, since a minimum in energy consumption results in maximizing the execution time

$$\underline{c_j} = \frac{C_j}{min(d_j; l_n) - max(s_j; f_n) + 1}, \underline{c_j} \in \mathbb{R}^+ . \tag{3.1}$$

This model therefore sets a minimum flow time, and minimizes energy consumption from there. This ensures that the computing requirement of the job is stretched in a just-in-time manner, allowing for a degree of energy saving by reducing the processing speed, or voltage on the CPU. For this model it is assumed that using less than total capacity automatically powers down the CPU to meet the required amount of CPU cycles.

3.2 Optimal solution

Allocation is determined by the binary decision variable x_{jnt}, where $x_{jnt} = 1$ if job j is allocated to node n in time slot t, and $x_{jnt} = 0$ if not. The time horizon $T = t \in \mathbb{N}$ defines the set of all time slots for the underlying allocation problem. This ensures that no jobs or nodes are allocated in undefined time slots. Hence, if J is defined as the set of job requests (resource requests) and N is the set of defined resource nodes (node suppliers), the winner determination problem can be mathematically formalized as follows:

$$maxV_x := \sum_{j}^{J} c_j \sum_{n}^{N} \sum_{t}^{T} x_{jnt}(b_j - \epsilon_n) \tag{3.2}$$

$$s.t. \quad x_{jnt} \in \{0,1\}, \quad \sum_{n}^{N} x_{jnt} \leq 1 \quad \forall j \in J, \quad t \in T \tag{3.3}$$

$$\sum_{s_j}^{d_j} \sum_{n}^{N} x_{jnt} = (d_j - s_j + 1) \sum_{n}^{N} x_{jnt} \quad \forall j \in J, \quad t \in T \tag{3.4}$$

$$f_n \leq s_j \leq l_n, \quad f_n \leq d_j \leq l_n, \quad s_j \leq t_j \leq d_j \tag{3.5}$$

$$b_j \leq r_n \tag{3.6}$$

$$\sum_{j}^{J} x_{jnt} \underline{c_j} \leq c_n, \quad \forall n \in N, \quad \forall t \in T . \tag{3.7}$$

The objective function (3.2) maximizes the total welfare over all periods, jobs and available nodes. Welfare is in this case determined by the difference between the agents bid, and the energy required to complete the job. Constraint (3.3) introduces the binary decision variable x and ensures that a job can only be executed once on a single node. Together with (3.4) atomicity is enforced since the allocated duration must always equal the maximum possible duration. It is important to

Table 3. Sample allocation of jobs to nodes.

x	J1	J2	J3	J4	J5	J6
N1	1	0	0	0	1	0
N2	0	1	0	1	0	1
N3	0	0	0	0	0	0

note is that the scheduler provisions without preemption and once submitted, the jobs continue until completed. The inequalities in (3.5) ensure that a job can only be allocated to a node which is available during the time slots required while also ensuring that the reservation price of the node does not exceed the jobs willingness to pay (3.6). (3.7) is the restriction that the jobs allocated to a node are not allowed to consume more resources than are available on the node.

Example 5. Using distributions from Table 1 and 2, the optimized allocation, calculated using a General Algebraic Modeling System (GAMS)[1], amounts to a total welfare ($\sum_j^J \sum_n^N \sum_t^T x_{jnt} b_j$) of 3111 MU which includes energy costs of 1692 MU. For all intents and purposes, the pricing scheme to determine welfare is handled by using a first-price auction principle, which simplifies the calculations. This allows for direct comparison of different mechanisms. The sample allocation is shown in Table 3. It shows that J1 and J5 are processed on N1, while J2, J4 and J6 are processed on N2, with N3 left empty and J3 not processed.

[3] found that proposing an efficient solution that maximizes social welfare, and which yields incentive compatible prices is computationally intensive, requiring the solution of multiple instances of an NP-hard (nondeterministic polynomial-time hard) problem. Solving this winner determination problem with a perfect optimal solution makes scheduling in short intervals of a matter of seconds impossible, as in practice users require the resources in a timely manner. Given this NP-hard problem, [29] propose an allocation scheme based on a heuristic solution, since markets generally require fast, real-time solution techniques. They present such a heuristic allocation scheme for clearing large-scale Grid environments. For this model, their heuristic is extended to the setting where the welfare, as a function of bids and energy costs, is maximized in a green allocation scheme. The basic idea behind the model, and therefore its heuristic solution, is that by sorting the jobs and nodes with respect to their bids and energy efficiency, we allocate important jobs which yield highest revenue to efficient nodes, and use inefficient nodes last. This green heuristic scheme is presented in the following section.

3.3 Green heuristic

The goal of this model remains minimizing the energy consumption. Therefore wherever feasible the efficient nodes will always be preferred and if possible the

[1] available at http://www.gams.com

inefficient nodes will be kept powered down. Thus it is essential for some degree of energy saving to occur, that over all nodes, the output is less than 100%, i.e. less demand than supply.

Proposition 3.2. *In the case of excess demand, the green heuristic algorithm is no more energy efficient than any other heuristic algorithm.*

Proof. In attempt to conserve energy, inefficient nodes will be kept powered down. Given two nodes, one with high energy costs and another with low energy costs, the optimal solution would be to use the node with low energy costs as much as possible. If however both nodes need to run at full capacity, because of excess demand, no energy conserving potential exists, pending threshold selection. □

A more realistic scenario is the case of excess supply. With more nodes available than required to process all jobs, idle processors are quite common. The scheduler therefore chooses to either power down all nodes except the most efficient one and run all jobs on that one node, or spread out the jobs to all nodes evenly, running all nodes at a constant rate, or a combination of both. Since energy consumption is exponentially related to processing speeds, running all jobs on a single node is inefficient, which points towards spreading the jobs over all nodes. However, there are fixed energy costs involved to simply run a system, regardless of processing on it or not. Therefore the optimal solution lies somewhere in between, keeping some nodes powered down, and spreading the jobs as much as possible on the remaining available nodes. Figure 1 shows a flowchart of the procedure, describing:

1. For each time slot, the jobs $j \in J$ still in the queue are sorted in decreasing order of their bids b_j, and the nodes $n \in N$ in increasing order of their reported efficiency ϵ_n.
2. The scheduler then sequentially runs through the job ranking, starting with the highest valued job and tries to allocate this job to the most efficient, feasible node which still has idle capacity left. By sorting the jobs in this way, the difference $(b_j - \epsilon_n)$ is minimized.
3. For each time slot, the jobs currently in the scheduler are stretched so that their estimated finish time is the required deadline, freeing up CPU's and effectively allowing the processor to run at a lower speed, conserving energy.

Example 6. Table 4 shows the sample heuristic allocation resulting in a total welfare of 2888 MU which includes energy costs of 1618 MU. The same jobs are allocated, as in the optimal case, but the welfare is slightly less. In terms of competitive ratios, if V^* is the optimal efficient solution, and V^{green} is the solution generated by the green heuristic, then:

In terms of competitive ratios, if V^* is the optimal efficient solution, and V^{green} is the solution generated by the green heuristic, then:

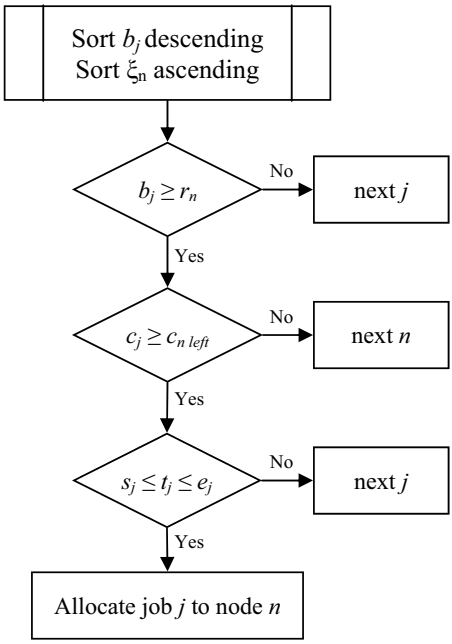

FIGURE 1. Heuristic allocation mechanism.

TABLE 4. Sample heuristic allocation of jobs to nodes.

x	J1	J2	J3	J4	J5	J6
N1	0	0	0	1	0	1
N2	1	1	0	0	1	0
N3	0	0	0	0	0	0

Proposition 3.3. *In the worst case, the performance ratio V^{green}/V^* can become arbitrarily close to zero.*

Proof. Consider the case of only a single period of allocation with only a single node, and two jobs which need to be allocated. Suppose the node offer has the tuple $(r, \epsilon, c) = (1\ MU, 1\ MU, \infty)$, and the two jobs have (b_j, c_j) $J1(3\ MU, 1)$ and $J2(2\ MU, z)$ respectively. The heuristic would for all z always allocate J1, generating profits and a welfare valuation of 2 MU, while the optimal solution, for $z > 3$ would be to allocate J2. As a result for $z \to \infty$, $V^{green}/V^* \to 0$. □

Although this is clearly an argument against the use of this heuristic, this is normally the case for heuristic solutions as they are a simplification of the

Table 5. Welfare and energy cost summary.

Algorithm	Welfare	Energy Costs
Optimal	100%	100%
Green	89%–96%	96%–104%
Greedy (Stoesser, 2007)	92%–99%	102%–112%

optimal case, trading some variable size, in this case processing time, at the cost of efficiency. Generally, the welfare solutions produced by heuristic allocations are lower than optimal solutions calculated with solvers. In some simulations however, the performance ratio (heuristic solutions against the optimum on average) reached nearly 93%.

It is expected, and will be shown, that the optimal solution yields the highest welfare, and the green heuristic has a slightly lower W/E-ratio than the optimal solution. Table 5 shows the summary of some generated results, using the settings drawn from Table 1, and optimized using GAMS:

Apart from computational speed and approximate efficiency, the heuristic allocation needs to be tested for other strategic properties inherent in the system, which could limit the potential gain of agents to misrepresent resource requirements. These will be discussed in Section 4.

4. Strategic incentives

In this section, strategic incentives will be analyzed, and how they affect the outcome of the schedule. Incentive compatibility is important in mechanism design as without it, the mechanism could potentially fail to allocate resources efficiently [2]. These incentives can be split into two categories: Incentives to misrepresent the resource characteristics, and incentives to misrepresent valuations.

4.1 Misreporting resource characteristics

When looking at incentives to misreport resource characteristics, these 'characteristics' include physical sizes which can be measured on a scale. These include computing requirements, and sizing of jobs. Proving that misreporting of computing requirements is no strategy is elementary, since if a job's resource requirement is understated, the job will in some cases not fit the schedule in the operational phase, implying that the agent has to pay for resources that are of no use to him. Similarly, overstating the resource costs are no incentive, as the agent has to pay for the job not fitting the schedule in the optimization phase. [15] addresses two further strategic properties of mechanisms, with regard to incentives of users to split or merge jobs (nodes). A mechanism is said to be split-proof, if the users cannot gain from splitting their jobs (nodes) into several smaller jobs (nodes).

A mechanism is said to be merge-proof if users cannot benefit from merging several jobs (nodes) to one bigger job (node).

Proposition 4.1. *The green heuristic is split-, and merge-proof.*

Proof (*Sketch*). To prove proposition 4, split- and merge-proofness first need to be defined:

- A mechanism is said to be *split-proof*, if the users cannot gain from splitting their jobs (nodes) into several smaller jobs (nodes).
- A mechanism is said to be *merge-proof* if users cannot benefit from merging several jobs (nodes) to one bigger job (node).

The sorting in Step 1 of the heuristic depends on the valuations per unit of computing power for the jobs, and the energy cost per CPU for the nodes. Merging and splitting simply changes the units of computing power required. With respect to the positioning within the ranking queue however, merging or splitting requests or offers does not affect the rankings. □

As a result, the strategy space for selfish users is restricted to misreporting their valuations only, by misstating the valuations or energy costs.

4.2 Misreporting valuations

The first involves the possible incentive to misreport the energy costs, i.e. report a high efficiency rate (low energy costs) to receive more jobs. The second is the possible incentive of agents to misreport their valuations by changing their bids. The third involves changing the deadlines of projects. These incentives are important, as their uncontrolled influence could cause the model to fail.

4.2.1 Misreporting energy costs

It is important for this model that the suppliers correctly report their energy consumption per unit of CPU. Previously, δ_n was assumed to be a markup, or error margin in estimating the energy costs in processing. By design, the node with the lowest energy consumption will receive the most jobs. With this allocation of jobs to nodes being heavily dependent on the energy costs of the nodes, the suppliers may have an incentive to misrepresent their energy costs by lowering its importance in the reservation price. This incentive could destroy the whole aim of this model to conserve energy. Artificially raising the energy costs in the reservation price is no strategy, as it only decreases the chance of a node being accepted, and therefore is no incentive problem.

In this model, the interdependency between δ_n and ϵ_n is assumed to be linear. Decreasing ϵ_n in $r_n = \epsilon_n + \delta_n$ they increase the probability that the node will be chosen before another node. In the worst case, a very inefficient node with high energy consumption could set ϵ_n very low, and compensate by increasing δ_n.

Proposition 4.2. *Agents have an incentive to undervalue their energy costs.*

Proof (*Sketch*). By mechanism design, there is no restriction on the suppliers to truthfully report his valuation. The node with the lowest energy costs receives the first priority, a clear possible incentive to understate their energy costs, since the energy costs only affect their local efficiency, and not the payments to the scheduler. □

There are however ways to counter this incentive. One possible strategy to ensure truthful submittal of energy costs lies in control, by monitoring the energy consumed by a system, and reporting it back to the user, or scheduler, as part of the grid operating system necessary to connect to the cloud. This in turn gives the scheduler the exact energy costs per unit of CPU. Coupled with a penalty system, an agent who misreports his energy costs could be fined a penalty after job procession, or even expelled from the grid, should the energy costs be higher than reported. Problematic with this method is that there is no way to discern between a δ_n-marginal error assumed in the model, or a strategic misreporting of δ_n^{mis}. The scheduler should therefore set some sort of boundary by which the misreported margins are clustered into 'truthful' errors and strategic behavior. The problem with this method is, that once these boundaries are known by speculating players, statistic evaluations will show that an unusual amount of reported errors will tend toward this margin. However, this over- or undervaluation is only profitable in the first trading period, since from the next round onward, the energy cost of the system is known.

4.2.2 Misreporting bids

Pricing has always been difficult in new markets, as in the case for Grid resources. Since jobs generally tend to be unique, there is no real "learning" available for price setters. Auction mechanisms allow market makers to determine the preferences of buyers, by asking for a bid. In most cases where consumers are required to report their willingness to pay, strategic behavior plays a vital role in pricing the final outcome. In this model, the agents may have an incentive to set their bids arbitrarily high, since reporting a high bid increases the probability that their job is scheduled. This would result in everyone quoting a bid higher than their true valuation v_j, if the price for the nodes remain constant.

In all games, where agents simply place a bid paying a constant price, all agents will choose an arbitrarily high number, since the bid has no influence on the actual payment. By forcing the agents to pay their bid however, a norm in a first price auction setting, the bids placed by the agents generally tend to be different from their true valuations, resulting in all agents bidding more aggressively than in the symmetric setting, where all prices and bids are known to all. A solution could be a second-price auction, or Vickrey auction setting, where the winner pays the price of the runner-up bid in the queue. Unique to this type of auction is its

incentive compatible mechanism design which imposes a strategy to all bidders to bid their true valuations with at least weak dominance. (cf. [32]) As a result, agents may have an incentive to misreport their bids in hope that their jobs will be chosen first.

Proposition 4.3. $\forall b_j^{'} > v_j,\ U(b_j^{'}) \leq U(v_j)$

Proof (*Sketch*). If an agent decides to bid over his valuation it only increases his probability of winning the auction and with it the probability that another bidder may bid higher than his own true valuation. This could result in a loss for the agent, since the price he has to pay will be higher than his valuation with a positive probability. For example if 2 bidders value the CPU's at $b_1 = v_1 = 8\ MU$ and $b_2 = v_2 = 10\ MU$, bidder 2 would win the auction, paying 8 MU and earns a surplus of 2 MU. If bidder 1 then bids 11 MU in order to win the auction, and has to pay 10 MU. Even if he bids 100 MU, the price will be the same; however, since his true valuation remains 8 MU, he will always pay 10 MU leaving him with a surplus of $-2\ MU$. □

Therefore, $U(b_j > v_j) \leq U(b_j = v_j)\ \forall b_j < b_{-j}$.

Agents therefore have no incentive to misreport their bids in hope that their jobs will be chosen first as a result of the VCG second price auction system.

4.2.3 Misreporting deadlines

Agents may have an incentive to misreport their deadlines, to increase their utility which could have dire consequences for the energy efficiency of the grid environment. Deadlines are essential for determining the CPU's required per time slot, and inadvertently the processing speed of the node it is scheduled on. Let t be the first time slot, $(t + k)$ the deadline, and m the margin by which the agent misreports his deadline.

Proposition 4.4. $\forall d_j^{'} \neq d_j^{true},\ U(d_j^{'}) \leq U(d_j^{true})$

Proof (*Sketch*). The speed at which a job is executed is directly dependent on its deadline. An agent who submits his job in time slot t, and reports a deadline in time slot $(t+k)$, will receive the job in time slot $(t+k)$ and no earlier. By instead reporting a deadline as time slot $(t + k - m)$, the agent influences the processing speed $\frac{C}{k-m+1}$ at which the same job is executed, and his bid, based on the CPUs used per time slot. Effectively his bid would be higher than originally, since his overall valuation of the job remains the same but the CPUs required per time slot increases. By assumption the agent whose true deadline remains in time slot $(t+k)$ is indifferent between a job handed to him in time slot $(t+k-m)$ or $(t+k)$. However, the price he has to pay, and thereby his bid increases. In the minimal case, if the bid remains the same, $\forall d_j^{'} < d_j{}^{*},\ U(d_j^{'}) = U(d_j{}^{*})$. Likewise to report a later deadline than his true time is no option, as the job will be delivered too

late, given that the deadline can be compared to a final delivery date this would result in $\forall d_j^{'} > d_j^*$, $U(d_j^{'}) = 0$. □

Therefore, as long as the agents true deadline does not change by reporting a different deadline, the agent has no incentive to misreport his deadlines.

5. Conclusion

To conclude this work, the research question 'Can the current scheduling models be altered in that they allow for more energy efficiency in the Grid environments, without loss of incentive compatibility even if not all valuations are known?' can only be confirmed. This work proposed the use of an energy based scheduling heuristic for Grid applications based on system-centric models common to the current approach to allocating Grid resources. While we only looked at the scheduling of CPU power, the model can easily be extended to include varying voltages or more sophisticated power models into the power consumption function including peripheral devices like in [8], or [37], or to include memory requirements common in models for large data centers with storage intensive jobs. Also this work presents a mechanism which achieves a distinct trade-off between allocative efficiency and energy efficiency, computational tractability and incentive compatibility in a simple theoretic model, followed by a detailed analysis, yielding insight to favorable incentive properties, or at the least presents solutions to counter non-favorable ones.

It should be noted that the model presented in this work is a strong simplification of a real-world Grid setup. Provisioning was isolated and a model built around it. In reality, work flows are too complex and interdependent in order for scheduling to be done using such isolated means. Nonetheless, it still outlines basic design options of how the mechanisms can be integrated into current Grid schedulers. Business processes for example are often very volatile, in that they change as the environment of the business process changes. Especially when looking at business processes or business work flows, the realization or computation of a 'perfect' schedule is far more complex than shown here. For example, some jobs may have to be executed monthly, like payrolls; others are one-shot calculations. In further research, we intend to:

- Include cooling costs explicitly by incorporating temperature as an important factor.
- Use larger real workload traces to demonstrate the goodness of our optimization procedure. (Feitelson, 2002)

References

[1] S. Albers and H. Fujiwara: *Energy-Efficient Algorithms for Flow Time Minimization.* ACM Transactions on Algorithms 3 (4) (2007), Article 49, pp. 1–16.

[2] G. A. Akerlof: *The Market for 'Lemons': Quality Uncertainty and the Market Mechanism.* Quarterly Journal of Economics 84 (3) (1970), pp. 488–500.

[3] R. Bapna, S. Das, R. Garfinkel and J. Stallaert: *A Market Design for Grid Computing.* INFORMS Journal on Computing 20 (1) (2008), pp. 100–111.

[4] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt and A. Warfield : *Xen and the art of Virtualization.* ACM SIOPS Operating Systems Review 37 (5) (2003), pp. 164–177.

[5] B. Strassmann: *Der Fußabdruck des Surfers.* Die Zeit 33 (2007) p. 29.

[6] R. Buyya, D. Abramson and S. Venugopal: *The Grid Economy.* Proceedings of the IEEE 93 (3) (2005), pp. 698–714.

[7] B. N. Chun and D. E. Culler: *Market-based proportional resource sharing for clusters.* Millenium Project Research Report (1999), http://www.cs.berkeley.edu/~bnc/papers/market.pdf.

[8] A. K. Coskun, T. S. Rosing, K. A. Whisnant and K. C. Gross: *Temperature-Aware MPSoC Scheduling for Reducing Hot Spots and Gradients.* Design Automation Conference, 2008. ASPDAC 2008. Asia and South Pacific, pp. 49–54.

[9] EPA: *Building a Powerful and Enduring Brand.* http://www.energystar.gov/ia/partners/downloads/ENERGY_STARBndManf508.pdf – 28.04.2008.

[10] T. Eymann, S. Sackmann and G. Müller: *Hayeks Katallaxie – Ein zukunftsweisendes Konzept für die Wirtschaftsinformatik?* Wirtschaftsinformatik 45 (5) 2003, pp. 491–496.

[11] D. G. Feitelson: *Workload modeling for performance evaluation.* In: Performance Evaluation of Complex Systems: Techniques and Tools, M. C. Calzarossa and S. Tucci (Eds.), Springer, London. Lecture Notes in Computer Science 2459 (2002), pp. 114–141.

[12] I. Foster and C. Kesselman: *The Grid: Blueprint for a New Computing Infrastructure.* Morgan Kaufmann Publishers, San Francisco, 1999.

[13] I. Foster, C. Kesselman and S. Tuecke: *The Anatomy of the Grid.* In: International Journal of Supercomputer Applications, 2001.

[14] M. Landsberger, J. Rubinstein, E. Wolfstetter and S. Zamir: *First-price auctions when the ranking of valuations is common knowledge.* Review of Economic Design 6 (2001), pp. 461–489.

[15] H. Moulin: *Proportional scheduling, Split-proofness, and Mergeproofness*, Games and Economic Behavior 63 (2) (2008), pp. 567–587.

[16] R. B. Myerson, *Game Theory: Analysis of Conflict*, Harvard University Press, Cambridge MA, 1991.

[17] D. Neumann, S. Lamparter and B. Schnizler: *Automated Bidding for Trading Grid Services.* Proceeding of the European Conference on Information Systems (ECIS), 2006.

[18] D. Neumann, J. Stößer, C. Weinhardt and J. Nimi: *A Framework for Commercial Grids – Economic and Technical Challenges.* Journal of Grid Computing 6 (2008), pp. 325–347.

[19] D. C. Parkes, J. Kalagnanam and M. Eso: *Achieving Budget-Balance with Vickrey-Based Payment Schemes in Combinatorial Exchanges*, IBM Research Report RC 22218 W0110–065 – see also: Proc. 17th International Joint Conference on Artificial Intelligence (IJCAI-01) (2002), pp. 1161–1168.

[20] C. Pettey: *Gartner Estimates ICT Industry Accounts for* 2 *Percent of Global* CO_2 *Emissions.* http://www.gartner.com/it/page.jsp?id=503867 – 28.04.2008.

[21] K. Pruhs, P. Uthaisombut and G. Woeginger: *Getting the best response for your ERG.* In: Proceedings of the 9th Scandinavian Workshop on Algorithm Theory (SWAT), Lecture Notes in Computer Science 3111 (2004), pp. 15–25.

[22] L. Rasmusson: *Network capacity sharing with QoS as a financial derivative pricing problem: algorithms and network design*, PhD thesis, Universitet Stockholms, 2002.

[23] O. Regev and N. Nisan, *The POPCORN Market – An Online Market for Computational Resources.* Proceedings of the 1st International Conference on Information and Computational Economies (1998), pp. 148–157 – see also: Decision Support Systems 28 (1–2) (2000), pp. 177–189.

[24] C. Rusu, R. Melhem and D. Moss: *Maximizing Rewards for Real-Time Applications with Energy Constraints.* ACM Transactions on Embedded Computer Systems 2 (4) (2003), pp. 537–559.

[25] See: *Is there a pathway to a Green Grid.* http://www.ibergrid.eu/2008/presentations/Dia.

[26] B. Schnizler, D. Neumann, D. Veit and C.Weinhardt: *Trading grid services – A multi attribute combinatorial approach.* European Journal of Operational Research 187 (3) (2008), pp. 943–961.

[27] S. Shivle, H. J. Siegel and A. A. Maciejewski et al.: *Static allocation of resources to communicating subtasks in a heterogenous ad hoc grid environment.* Journal of Parallel and Distributed Computing 66 (4) (2006), pp. 600–611.

[28] Singh, Hayward and Anderson: *Green IT takes Center Stage.* Springboard Research (2007).

[29] J. Stößer, D. Neumann and C. Weinhardt: *Market-Based Pricing in Grids: On Strategic Manipulation and Computational Cost.* Journal of AIS Sponsored Theory Development Workshop, (2007).

[30] TCO: *TCO labelling produces results.* http://www.tcodevelopment.com/pls/nvp/Document.Show?CID=1200\&MID=34 – 28.04.2008.

[31] tecChannel: *Umfrage: GreenIT hat Zukunft.* http://www.tecchannel.de/482677 – 28.04.2008.

[32] H. R. Varian: *Economic Mechanism Design for Computerized Agents* in Proceedings USENIX Workshop on Electronic Commerce 1995. Minor update 2000.

[33] H. R. Varian: *The Information Economy: How much will two bits be worth in the digital marketplace?* Educom Review 31 (1) (1996), pp. 44–46.

[34] C. A. Waldsburger, T. Hogg, B. A. Huberman, J. O. Kephart et al.: *Spawn: A Distributed Computational Economy*, IEEE Transactions on Software Engineering (18) 2 (1992), pp. 103–117.

[35] M. Weiser, B. Welch, A. Demers and S. Shenker: *Scheduling for reduced CPU energy.* In: Proceedings of the Symposium on Operating Systems Design and Implementation (1994), pp. 13–23 – see also: The International Series in Engineering and Computer Science 353, Springer US, 1996, pp. 449–471.

[36] R. Wolski, J. S. Plank, J. Brevik and T. Bryan: *Analyzing Market-Based Resource Allocation Strategies for the Computational Grid.* International Journal of High Performance Computing Applications 15 (3) (2001), pp. 258–281.

[37] F. Yao, A. Demers and S. Shenker: *A Scheduling Model for Reduced CPU Energy.* Foundations of Computer Science, 1995. Proceedings 36th Annual Symposium (1995), pp. 374–382.

Christian Bodenstein
Chair for Information Systems Research
Albert-Ludwigs-Universität Freiburg
Platz der Alten Synagoge
79085 Freiburg
Germany
e-mail: `christian.bodenstein@is.uni-freiburg.de`

Economic Models and Algorithms for Distributed Systems, 257–269
Book Series: Autonomic Systems

Facing Price Risks in Internet-of-Services Markets

Raimund Matros, Werner Streitberger, Stefan Koenig and Torsten Eymann

Abstract. Internet-of-Services markets allow companies to procure computational resources and application services externally and thus to save both internal capital expenditures and operational costs. Despite the advantages of this new paradigm only few work has been done in the field of risk management concerning Internet-of-Services markets. We simulate such a market using a Grid simulator. The results show that market participants are exposed to price risk. Based on our results we identify and assess technical failures which could lead to loss on service consumer's side. We also show that technical failures influence service prices which lead to volatile prices. Both, service provider and service consumer are exposed to this uncertainty and need a way to face it. Therefore we apply a financial option model to overcome price risk.

Mathematics Subject Classification (2000). Primary 68Q10; secondery 68Q85.

Keywords. Grid computing, scheduling, heuristic, greenIT

1. Introduction

Businesses have to encounter several different challenges, when it comes to using Information Technology (IT). The increasing dynamism of markets leads to a continuous need for IT-Business-Alignment and the control of IT investments and resources. For every-day business, the use of computationally intensive IT seems essential to implement new flexible business models within a short time. In contrast to these advantages, the operational expenses of the technology, including usage and maintenance of the IT-infrastructure, are exploding; in particular, if the resources in question, like storage or cpu power, need to be dimensioned to cover

peak demand while only sparsely used otherwise. Staying competitive requires saving costs in this area [1].

The Internet-of-Services describes a general computational paradigm, which allows companies to procure computational resources and application services externally and thus to save both internal capital expenditures and operational costs. Depending on how the resources are traded and who the external provider is, several sub-concepts can be distinguished. The notion of Cloud Computing follows the idea of consuming different services externally, not from a distinct service provider, but from a blurred cloud of resources within a single business unit or even between different businesses [2]. Utility Computing emphasizes the similarity of procuring computational power like water or electricity seamlessly from a public infrastructure like the Electricity Grid [3] only when needed.

For the provider of Internet-of-Services, the business model lies in the economies of scales. From a technical point of view, Internet-of-Services virtualizes physical resources to become logical units, which can be assigned to different users. Then, using resources in parallel becomes possible, leading to overall better utilization and the execution of computationally intensive jobs within shorter time. One important characteristic is the distributed, perhaps redundant provision of storage, processing power, or more abstract services that extend over different organizations [4,5]. The heterogeneity of services and resources is opaque to the end user, who transparently uses a homogeneous service supply. An efficient allocation mechanism between service demand and supply is needed to get such an environment running – a market. The idea of applying markets to distributed systems is rather old [4–6], but leads to challenging problems, e.g. the moral risk the market participant has to deal with. In addition, both transaction participants deal with uncertainty caused by environmental factors (e.g. network failures) and problems determined by the markets themselves, i.e. price risks. We will investigate these risk within the following chapter.

Emerging Cloud computing markets come along with imperfections, restrictions and risks that we already know from financial markets. However, there are differences between financial markets and Internet-of-Service economies. These characteristics have to be taken in to account to assessing risks by applying methods from finance to Cloud computing markets. The paper's aim is first to identify differences between financial and Cloud computing markets concerning the upcoming risks and then apply and adapt an option price model from financial markets.

The following sections of this paper are organized as follows: Section 2. presents background and related work. Section 3. describes simulation results and the application of the application of an option price model. Finally, in Section 4., we will summarize the findings of the paper and discuss future research directions.

2. Background and related work

2.1 Background of cloud computing markets

The cloud computing paradigm is not a new concept. According to our point of view Clouds are a logical evolution of the Grid concept and they must be built on top of Grids. For characterizing Clouds in the context of Grids we use a layered model based on [7–10] which is depicted in Figure 1. According to our model raw resources consists of storage capacity like hard disk drives or solid state disks and utility resources like energy, computation and network bandwidth. Utility resources reflect a vision where IT can be accessed in analogy to electricity or water [7, 11]. Raw resources can be combined to bundles which can be treated as virtual units (VU). These virtual units or virtual servers represent a basis for basic services.

Basic services provide elementary functionalities like security, database, transformation or accounting. Up to this layer we speak of a general purpose Grid. Any abstraction above general purpose Grids reduces system semantics. This reduction in complexity comes along with an increase of ease of use which is reflected by syntactically simplicity of interfaces [12]. In our comprehension this is what we call domain specific Grid. This classification can consist if infrastructure services like storage Clouds or compute Clouds. The vertical specialization increases bottom-up. Complex services which consists of services from subjacent layers can be described by using ontology-based frameworks [13]. In the paper's context we differ between software services, platform services and markets. We only concentrate on software and platforms to outline Cloud computing due to the absence of service markets up to now. There are several approaches to establish service markets but none has left the beta status yet[1]. In contrast to market places platforms constitute a way of mixing services up to enable service mash-ups.

Software as a Service (SaaS) providers are offering their software products in an internet environment that can be accessed at any time and from any computer. Providers either charge fees on a monthly or a pay per use basis. The service being sold is an end-user application which is restricted to what the application is and can do. Customers neither control nor know details of the underlying technology. These services can build on top of subjacent clouds but also can be offered standalone. This fact makes it difficult to separate Cloud software services from simple hosting services that run on dedicated servers or even run on the invoking host. Some of notable companies here are email providers or search engines. Platform as a Service (PaaS) is an outgrowth of the SaaS delivery model. The PaaS model offers all of the requirements to support the end-to-end life cycle of building and delivering web applications and services available from the internet. In contrast to services platforms are more flexible and enable compositions of services. Currently

[1]SORMA, a research project funded by the European Union, is going to build a platform where resource can automatically traded and Zimory (http://www.zimory.com) wants to build a commercial trading platform for IT services.

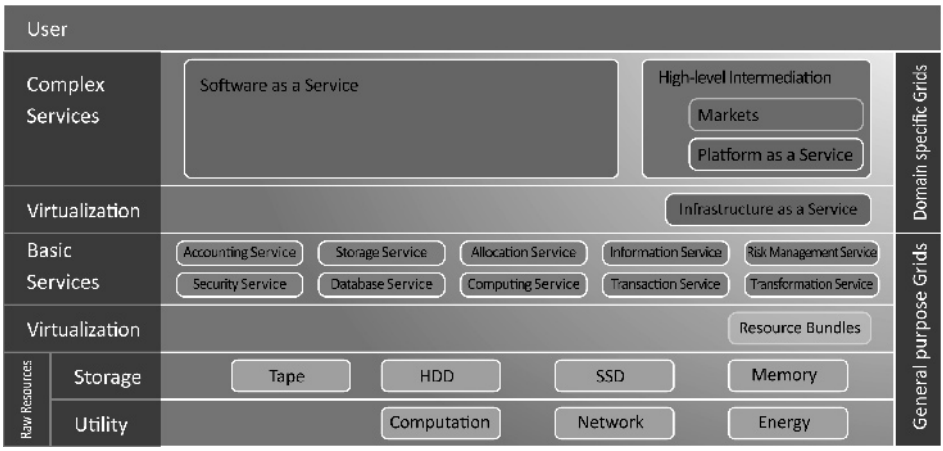

FIGURE 1. Classification of cloud components.

only a few competitors offer platform services[2]. Such high-level intermediation service providers can leverage Cloud infrastructures. Hence dissemination of Cloud platforms is in the interest of infrastructure providers.

Infrastructure as a Service (IaaS) is equivalent of SaaS for hardware devices. The customers pay to use shared infrastructures. Current payment models bring it to account on a monthly basis or a pay per use basis. Enterprises providing infrastructures enable Cloud services and Cloud platforms. Depending on the intended use there are storage Clouds like Amazon's S3 or computing Clouds like Amazon EC2. Mostly Cloud computing infrastructure providers additionally abstracts the basic services with some sort of server virtualization[3] [6]. SaaS, PaaS and IaaS can build on top of each other but this structure is not mandatory. Providers can play more than one role as well. Providers offering homogeneous services are sharing the same Cloud. Treating services in a Cloud as a commodity leads to the assumption of evolving service markets.

For the simulation of such a Cloud computing market, the paper uses the CATNETS Grid-Simulator [14]. The interdependencies between PaaS/SaaS and IaaS existing in Cloud computing market are separated by creating two interrelated markets: a resource market for trading of resource bundles; and a service market for trading software or platform services. This separation allows instances of a service to be hosted on different resources. In the simulation model, a Complex Service (CS) is a composite service, like a workflow, that requires the execution of other interdependent services, termed Basic Services (BSs). A CS is the entry point for the Cloud computing network. The traded products on the service market, the BSs, are completely standardized and have a single attribute name. The name is

[2]E.g. Google App Engine, Heroku, Mosso, Engine Yard or Salesforce
[3]Recent virtual machine technologies are Xen or VMWare

a unique identifier whose intended semantics is shared among all complex service providers. Multiple instances of the same BS can co-exist in the network. After a successful negotiation in the service market, BSs negotiate with Resource Providers (RPs) for the resources necessary to host services and serve the service requests. RPs utilize the existing resource management systems to allocate the necessary resources. RP order resources in Resource Bundles (RBs).

A resource bundle is described by a set of pairs of resource type and quantity. Every BS has an associated resource bundle. The bundle defines the type and quantity of resources needed for provisioning that service. In the CATNETS scenario, the resource bundle required for a BS is predefined for the sake of simplicity. In general, the model allows the use of any BS to resource bundle mapping function. In the resource market, the allocation process follows the service market. First, a Basic Service Provider (BSP) queries for RPs which are able to provide the specified resource bundle and ranks the received list of RPs according to the offered price. Second, the bargaining for the resource bundle is carried out. If the resource negotiation ends successfully, the BS is executed on the contracted resources from a RP.

2.2 Related work on upcoming risks

While some authors only focus on delivery risk when thinking of service markets [15], in our point of view this classification is too vague to understand all factors influencing uncertainty or risk. Therefore we have identified two different characteristics of risk in Cloud environments: the technical and the price risk.

2.2.1 Technical risk

In Service infrastructures, failures are the rule rather than the exception [16]. Most of these failures in current Service delivery environments rely on the technology, which includes the middleware and service itself, and the infrastructure, which includes the network and the Grid sites. As no measurement data from large-scale Infrastructures is available yet, risk studies of domain specific Grids are taken into account for risk identification and quantification. One reason for this lack of data is the Grid's organizational structure. Infrastructure providers aim to keep their business model secure. This incorporates the monitored data of failures concerning their provided services. The reliability of a Grid is significantly influenced by the following three risk categories [17]: Infrastructure failure: Sources for infrastructure failure are public networks like the Internet and provider's infrastructure itself. From a logical point of view, the sites are black boxes, which provide infrastructure services or higher-level services to customers. The risk associated with the execution of a job on resources depends on the time the resources are used by the job and the general availability of the infrastructure [17,18]. Higher risk comes from the network. Both, incoming and outgoing data transfer together with its amount can significantly influence the service quality [18]. The main reason is the

best-effort behaviour of public networks like the Internet. Beside the hardware, all software components can influence on the risks. Middleware toolkits and the offered services show faulty behaviour in failure studies [18, 19].

The *global resource management* itself is a source of failures. Inaccurate and outdated information about the resource status lead to misallocations by the selection and negotiation mechanism. There could also be the possibility that the allocation mechanism don't meet economic objectives therefore lead to inappropriate allocations. A lack of suitable resources can also increase the risk coming from the global resource management. Waiting queues: The last main risk source is waiting queues, which can cause missed time deadlines specified by the service customer. If there is highly fluctuating demand in the infrastructure Cloud, a job can wait for long time in a waiting queue until the job will be execution. This will lead to a failed job execution in the end.

2.2.2 Price risk

We define price risk as follows: The price risk in Internet-of-Service markets is the risk of price change. This change has its roots in explicit factors like fluctuating demand, resource prices and technological development as well as in implicit factors like inconsistent expectations, asymmetrical distributed information and expected technical risks. Market prices reflect these implicit and explicit influence factors. All these influence factors can lead to the conclusion that prices in services economies are not predictable. Price fluctuations can cause problems for both parties service providers and service consumers. The provider can suffer from falling prices while the consumer benefits and vice versa.

Financial markets show us how to manage price risk. The methodology of derivatives offers the opportunity to encounter price risk. According to Hull a derivative is a financial instrument whose value changes in response to the changes in underlying variables [20]. Trading Service derivatives requires an infrastructure which is able to handle advance resource reservations. Technical issues for resource reservation are discussed in [21, 22]. In the following we don't focus on technical premises. We mainly address the concept from which requirements for further implementations can be derived. There are only a few approaches which deal with derivatives in Internet-of-Service markets. Rasmussen and Petterson use a financial option approach [23, 24] which we propose as well. In contrast to their models our work includes technical risks influences on prices. Meinl uses a real option approach [25] derived from findings of [26].

Derivatives can be based on different types of variables. In our context, virtual units which are traded on spot markets represent the underlying variable for derivative instruments. The use of derivatives is to hedge risk for one party. That means a service buyer is able to hedge against rising prices by pegging the price for a service to a negotiated value for an execution in the future. This contract is called future. The provider is also able to hedge against falling prices by buying

short options which generate profit when prices fall. It is obvious that a future exchange opens the door for participants having different motivations. Service providers want to eliminate risk of decreasing prices while service consumers want to eliminate risk of rising prices. But intermediaries who could act as speculators could also enter the market with the intention to make profit.

3. Quantifying and overcoming risk

3.1 Simulating cloud computing markets

Our scenario is based on the service-oriented simulation environment used in [27] and the market structure denoted in [14]. Our environment represents a small network with 30 nodes. We consider natural distance between nodes by simulating latency as well as infrastructure failure. Each node is linked to at least two other neighbors (network degree ≥ 2). This setting is very similar to a small enterprise computing center. The market structure is described as follows: For our simulation we use 10 CSAs, 10 BSAs and 10 RAs. We focus on the service market where a homogenous resource bundle (virtual unit) is traded. The simulation model assumes that the Basic Service execution is always reliable. Only the service allocation mechanism faces failure during the allocation. Delayed messages on the network caused by waiting queues and network congestion influences the prices in two ways:

1. A Complex Service as a service consumer adapts to failures during the service allocation by increasing his price. He interprets a failed allocation as a reject from the provider, because he does not know whether there is a failure or the provider's answer message is delayed. As a consequence, he increases his reservation price for the next transaction. If a Complex Service allocates a Basic Service successful, he lowers his reservation price for the next transaction.
2. A Basic Service as a service provider acts in a similar way. A not responding Complex Service is interpreted as a failed transaction which leads to a decreasing price for his service. If there was a successful allocation of the provider's Basic Service, the Basic Service Provider increases his price for the next transaction.

Figure 2 illustrates a simulated price process on the service market over 1505 observations. The x-axis shows the time scale which is standardized to minutes. There can be easily seen that market prices are uncertain over the observed time period. We determine a standard deviation of price changes of $\sigma = 0.4295$ for a trading period of 400 minutes. This is the base risk metric for agents' future expectations. In the following we presume that market prices reflect all information available including infrastructure failure. This assumption implies the validity of the Efficient Market Hypothesis which explains how the price changes with chang-

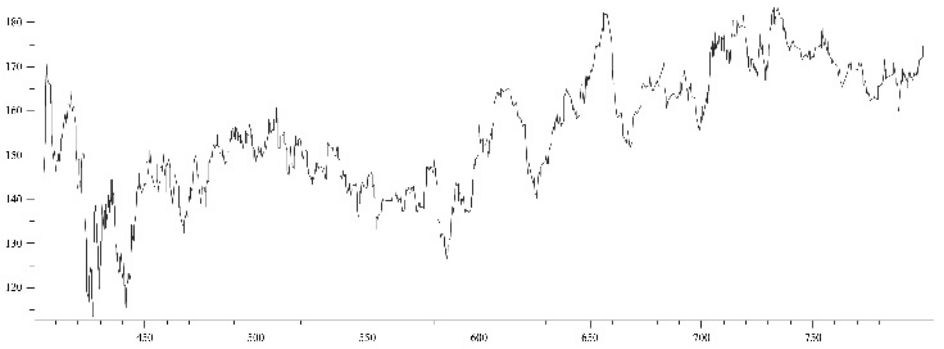

FIGURE 2. Simulated price process.

ing information [20]. Furthermore we also presume absence of arbitrage [4, 28] in our simulated market.

3.2 Applying an option price model to the simulated cloud computing market

In the following we build a mathematical model describing the price process of resource bundle trades in order to give agents the ability to forecast prices and price the risk. Therefore we propose a random walk model following the Markov property which is denoted as a stochastic process whereby the behavior of the variable over a short time depends merely on the value of the variable at the beginning of the period, not on its past history [20]. Applying a random walk involves two hypotheses: consecutive price changes are independent and changes in price follow some probability distribution [28]. As a further step we obtain a stochastic differential equation with μ as constant return, σ as volatility and W_t as stochastic process

$$dX(t) = \mu X(t)dt + \sigma X(t)dW_t\,. \tag{3.1}$$

We use a standard Wiener process to show continuous trajectories of 3.1. The Wiener process W_t is characterized starts at $W_0 = 0$ therefore W_t has independent increments with the distribution $W_{t_2} - W_{t_1} \approx N(0, t_2 - t_1)$. The condition that it has independent increments means that if $0 \leq s1 \leq t1 \leq s2 \leq t2$ then $W_{t_1} - W_{s_1}$ and $W_{t_2} - W_{s_2}$ are independent random variables.

Figure 3 shows a simulation of 7 stochastic trajectories with a starting value of 145 with a volatility of $\sigma = 0.4295$ over a period of 400 minutes. The thick line represents the benchmark of the multiagent simulation as a reference. In addition the grey area illustrates the mean deviation spread over time in 100 simulation runs which is similar to the results we gained in 100 multi agent simulation runs. Therefore we assume in the following an equal deviation spread.

We are now able to model the price process of virtual servers in a mathematical manner which makes a derivative pricing for advance reservations possible.

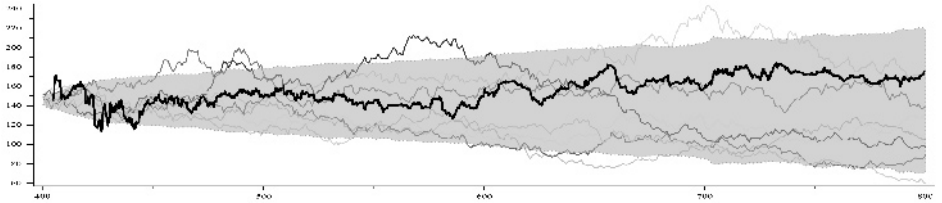

FIGURE 3. Simulated wiener process of virtual unit pricing.

For further derivative pricing we use the Black and Scholes model which can be applied due to the assumption that virtual unit prices follow a Wiener process [29]. According to Black and Scholes the valuation of options follows:

$$o(x,t) = x * N(d_1) - c * e^{r*(t-t^*)} \tag{3.2}$$

$$d_2 = \frac{ln(\frac{x}{c}) + (r + \frac{1}{2} * \sigma^2) * (t^* - t)}{\sigma\sqrt{t^* - t}} \tag{3.3}$$

$$d_1 = \frac{ln(\frac{x}{c}) + (r - \frac{1}{2} * \sigma^2) * (t^* - t)}{\sigma\sqrt{t^* - t}}\,. \tag{3.4}$$

The variables are described in the following example: An agent with a planning horizon of 400 minutes and a determined demand of 100 virtual units in $t^* = 650min$ wants to overcome price risk. The agent expects an increase in virtual unit's price. Therefore he buys 100 call options with strike $x = 61.23$. According to (3.2) the price of one option depends on the current service price x and the date of valuating the option t which results in $o(x,t) = 14.95$ with $x = 61.23$ and $t = 250$. At $t = 650min$ the value of one option equals to $o(x,t) = 38.72$, with $x = 101.81$. In this case the agent will exercise the options at maturity date because they are in money with a total gain of 2377. In this example we abstracted from an interest rate ($r = 0$) due to agent's short planning horizon ($t^* - t = 400$). $N(d_1)$ measures the sensitivity to changes in the price of a service and denotes the option's probability of ending in the money. Figure 4 illustrates prices of the option and the underlying virtual unit. If the agent would have supplied his demand on the spot market he had paid 10181.

4. Conclusion

Based on our findings by simulating an Internet-of-Service market we illustrated that this market is exposed to price risk. We identified and simulated technical failures which could lead to loss or damage on service consumer's side. We illustrated that technical failures influence service prices which make prices unpredictable. Both, service provider and service consumer are exposed to this uncertainty. To overcome risk of price change we propose the adoption of a financial

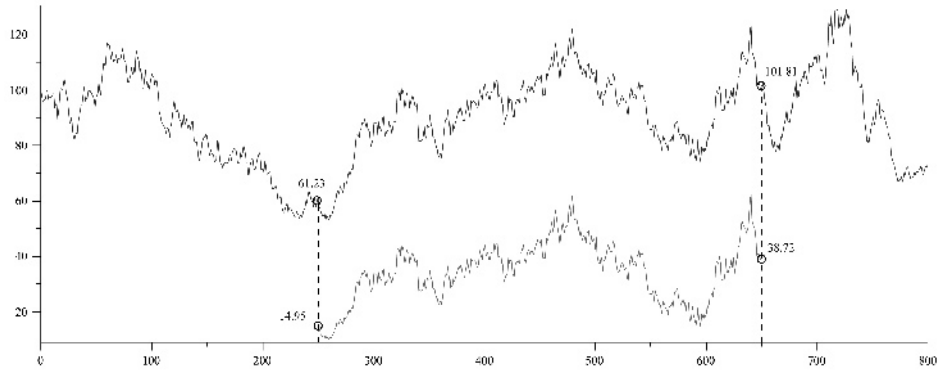

FIGURE 4. Option pricing example.

option approach. We showed that our simulated price process is very similar to a known stochastic process which makes financial option valuation methods applicable. Thus we were able to build a mathematical model which expressed the price process. An example showed the usage of options in uncertain Internet-of-Services markets. In this example we illustrated that market participants can hedge against increasing prices by buying options. Although lessons taught us that financial risk management methodology is applicable to Internet-of-Services markets there is a number of open questions which must be addressed in further research.

We didn't make any statements how derivative markets can influence spot market prices. To answer this question a combined spot and future Internet-of-Services market must be built and simulated. Furthermore we didn't separate technical risk from fluctuating demand. A separation opens the door for competitively risk management approaches like service insurances. Moreover we abstract from asymmetrical distributed information among agents which could cause principal agent conflicts.

Acknowledgements

This work was supported by the EC under the FP6 programme [eRep project CIT5-028575], [SORMA project IST-034286] and [CATNETS project IST-003769].

References

[1] L. L. de Orlov: Computing: Addressing today's business issues. Technical report, Deloitte (2006).

[2] A. Weiss: Computing in the clouds. ACM **11**(4) (2007) 16–25.

[3] I. Foster and S. Tuecke: Describing the elephant: the different faces of IT as service. Queue **3**(6) (2005) 26–29.

[4] I. Foster and C. Kesselman: The Grid: Blueprint for a New Computing Infrastructure. Morgan Kaufmann (2004).

[5] I. Foster, C. Kesselman and S. Tuecke: The anatomy of the grid: Enabling scalable virtual organizations. International Journal of High Performance Computing Applications **15**(3) (2001) 200.

[6] J. Staten: Is cloud computing ready for the enterprise? Technical report, Forrester (2008).

[7] B. Blau, C. Block and J. Stoesser: How to trade electronic Services? Current status and open questions. (2008).

[8] B. Blau and B. Schnizler: Description languages and mechanisms for trading service objects in grid markets. In: Multikonferenz Wirtschaftsinformatik (MKWI), München (2008).

[9] T. Eymann, M. Reinicke, W. Streitberger, O. Rana, L. Joita, D. Neumann, B. Schnizler, D. Veit, O. Ardaiz, and P. Chacin: Catallaxy-based grid markets. Multiagent and Grid Systems **1**(4) (2005) 297–307.

[10] W. Streitberger, T. Eymann, D. Veit, M. Catalano, G. Giulioni, L. Joita and O. F. Rana: Evaluation of economic resource allocation in application layer networks – a metrics framework. (2007).

[11] M. A. Rappa: The utility business model and the future of computing services. IBM Systems Journal **43**(1) (2004) 32–42.

[12] J. Shantenu, A. Merzky and G. Fox: Using clouds to provide grids Higher-Levels of abstraction and explicit support for usage models. internal-pdf://cloud-grid-saga-3552647680/cloud-grid-saga.pdf (July 2008).

[13] S. Lamparter, A. Ankolekar, R. Studer and S. Grimm: Preference-based selection of highly configurable web services. Proceedings of the 16th international conference on World Wide Web (2007) 1013–1022.

[14] T. Eymann, D. Neumann, M. Reinicke, B. Schnizler, W. Streitberger and D. Veit: On the design of a Two-Tiered grid market structure. In: Business Applications of P2P and Grid Computing, MKWI. (2006).

[15] G. A. Paleologo: Price-at-Risk: a methodology for pricing utility computing services. IBM Systems Journal **43**(1) (2004) 20–31.

[16] M. Baker, R. Buyya and D. Laforenza: Grids and grid technologies for wide-area distributed computing. Software-Practice and Experience **32**(15) (2002) 1437–1466.

[17] Y. S. Dai, Y. Pan and X. Zou: A hierarchical modeling and analysis for grid service reliability. IEEE Transaction On Computers (2007) 681–691.

[18] O. Khalili, J. He, C. Olschanowsky, A. Snavely and H. Casanova: Measuring the performance and reliability of production computational grids. International Conference on Grid Computing (GRID), IEEE Computer Society (2006).

[19] H. Li, D. Groep, L. Wolters and J. Templon: Job failure analysis and its implications in a large-scale production grid. Proceedings of the 2nd IEEE International Conference on e-Science and Grid Computing (2006).

[20] J. Hull: Options, futures, and other derivatives. Prentice Hall Upper Saddle River, NJ (2005).

[21] K. Dean and M. Mark: Advance reservation and Co-Allocation protocol for grid computing. (2005).

[22] W. Reinhardt: In: Advance reservation of network resources for multimedia applications. (1994) 23–33.

[23] P. Pettersson: Financial derivatives for computer network capacity markets with Quality-of-Service guarantees. Technical report, Stockholm (2003).

[24] L. Rasmusson: Network capacity sharing with QoS as a financial derivative pricing problem: algorithms and network designRoyal Institute of Technology, Stockholm. PhD thesis (2002).

[25] T. Meinl: Advance reservation of grid resources via real options. (2008).

[26] P. R. Kleindorfer and D. J. Wu: Integrating long-and Short-Term contracting via Business-to-Business exchanges for Capital-Intensive industries. Management Science **49**(11) (2003) 1597–1615.

[27] W. Streitberger, S. Hudert, T. Eymann, B. Schnizler, F. Zini and M. Catalano,: On the simulation of grid market coordination approaches. Journal of Grid Computing **6**(3) (2008) 349–366.

[28] E. F. Fama: The behavior of Stock-Market prices. Journal of Business **38**(1) (1965) 34.

[29] F. Black, M. Scholes: The pricing of options and corporate liabilities. Journal of Political Economy **81**(3) (1973) 637.

Raimund Matros
University of Bayreuth
Chair of Information Systems Management
95440 Bayreuth
Germany
e-mail: raimund.matros@uni-bayreuth.de

Werner Streitberger
University of Bayreuth
Chair of Information Systems Management
95440 Bayreuth
Germany
e-mail: werner.streitberger@uni-bayreuth.de

Stefan Koenig
University of Bayreuth
Chair of Information Systems Management
95440 Bayreuth
Germany
e-mail: stefan.koenig@uni-bayreuth.de

Torsten Eymann
University of Bayreuth
Chair of Information Systems Management
95440 Bayreuth
Germany
e-mail: torsten.eymann@uni-bayreuth.de