Future Generation Computer Systems 28 (2012) 39–47

Contents lists available at SciVerse ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Cost–benefit analysis of an SLA mapping approach for defining standardized Cloud computing goods

Michael Maurer [a,*], Vincent C. Emeakaroha [a], Ivona Brandic [a], Jörn Altmann [b]

[a] Vienna University of Technology, Vienna, Austria

[b] Technology Management, Economics, and Policy Program & Department of Industrial Engineering, College of Engineering, Seoul National University, South Korea

ARTICLE INFO

Article history:
Received 25 January 2011
Received in revised form
12 May 2011
Accepted 28 May 2011
Available online 15 June 2011

Keywords:
Service level agreements
Cloud architecture
Cloud markets
Market liquidity
Cost modeling
Utility modeling
SLA matching
Goods standardization

ABSTRACT

Due to the large variety in computing resources and, consequently, the large number of different types of service level agreements (SLAs), computing resource markets face the problem of a low market liquidity. Restricting the number of different resource types to a small set of standardized computing resources seems to be the appropriate solution to counteract this problem. Standardized computing resources are defined through an SLA template. An SLA template defines the structure of an SLA, the service attributes, the names of the service attributes, and the service attribute values. However, since existing research results have only introduced static SLA templates so far, the SLA templates cannot reflect changes in user needs and market structures. To address this shortcoming, we present a novel approach of adaptive SLA matching. This approach adapts SLA templates based on SLA mappings of users. It allows Cloud users to define mappings between a public SLA template, which is available in the Cloud market, and their private SLA templates, which are used for various in-house business processes of the Cloud user. Besides showing how public SLA templates are adapted to the demand of Cloud users, we also analyze the costs and benefits of this approach. Costs are incurred every time a user has to define a new SLA mapping to a public SLA template due to its adaptation. In particular, we investigate how the costs differ with respect to the public SLA template adaptation method. The simulation results show that the use of heuristics within adaptation methods allows balancing the costs and benefits of the SLA mapping approach.

1. Introduction

Allocation of Cloud computing resources is based not only on functional requirements but also on different non-functional requirements. Non-functional requirements, e.g., application execution time, reliability, and availability, are termed as quality of service (QoS) requirements and are expressed by means of service level agreements (SLAs). In order to facilitate SLA creation and SLA management, SLA templates have been introduced. SLA templates represent popular SLA formats. They comprise elements such as names of trading parties, names of SLA attributes, measurement metrics, and attribute values [1].

Despite the existence of SLAs, buyers and sellers of computing resources face the problem of varying definitions of computing resources in Cloud computing markets. Computing resources are described through different non-standardized attributes, e.g., CPU cores, execution time, inbound bandwidth, outbound bandwidth, and processor type [2]. Sellers use them to describe their supply of resources. Buyers use them to describe their demand for resources. As a consequence, a large variety of different SLAs exists in the market. The success of matching offers from sellers and bids from buyers becomes very unlikely, i.e., the market liquidity (the likelihood of matching offers and bids) becomes very low [1].

Approaches that tackle this plethora of SLA attributes include the use of standardized SLA templates for a specific consumer base [3,4], downloadable predefined provider-specific SLA templates [5], and the use of ontologies [6,7]. These approaches clearly define SLA templates and require users to agree a priori on predefined requirements. These SLA templates are static meaning that they do not change nor adapt over time.

Consequently, the existing approaches for the specification of SLA templates cannot easily deal with demand changes. Demand changes of users are caused through different factors (e.g., changing market conditions). For example, the emergence of multi-core architectures in computing resources required the inclusion of the new attribute "number of cores", which was not present in an SLA template a couple of years ago. The existing approaches for the specification of SLA templates involve heavy user-interactions to adapt existing SLA templates to demand changes.

* Corresponding author.
E-mail addresses: maurer@infosys.tuwien.ac.at (M. Maurer), vincent@infosys.tuwien.ac.at (V.C. Emeakaroha), ivona@infosys.tuwien.ac.at (I. Brandic), jorn.altmann@acm.org (J. Altmann).

0167-739X/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.future.2011.05.023

In this paper, we apply adaptive SLA mapping, a new, semi-automatic approach that can react to changing market conditions [1]. This approach adapts public SLA templates, which are used in the Cloud market, based on SLA mappings. SLA mappings, which have been defined by users based on their needs, bridge the differences between existing public SLA templates and the private SLA template, i.e., the SLA template of the user. In our context private templates do not necessarily imply that they are inaccessible to others, but the word "private" is used to differentiate it from the "public" template of the (public) registry. So, all consumers' and providers' templates are called "private", whereas the registry's template is called "public". Since a user cannot easily change the private SLA template due to internal or legal organizational requirements, an SLA mapping is a convenient workaround.

Our adaptive SLA mapping approach can use different adaptation methods. The benefit of using an adaptation method is decreased by some cost for the user. Costs are only incurred, if a user has to define a new SLA mapping to a public SLA template due to its adaptation. Within this paper, we investigate these costs. In particular, we investigate how public SLA templates can be adapted to the demand of Cloud users and how the costs and benefits differ with respect to the public SLA template adaptation method used.

After introducing a reference adaption method for our analysis, we compare two additional adaptation methods which differ in the heuristics applied. The heuristics have been introduced in order to find a balance between the benefit of having a public SLA template that is identical to most of the private SLA templates and the cost of creating new SLA mappings and new public SLA templates. As the metrics for assessing the quality of the adaptation method, we define the overall system net utility of all users. The net utility considers the benefit of having the same attribute and attribute name in the public SLA template as in the private SLA template, as well as the cost of defining a new SLA attribute mapping.

The benefits of the adaptive SLA mapping approach for market participants are threefold. Firstly, traders can keep their private templates, which are required for other business processes. Secondly, based on their submitted mappings of private SLA templates to public SLA templates, they contribute to the evolution of the market's public SLA templates, reflecting all traders' needs. Thirdly, if a set of new products is introduced to the market, our approach can be applied to find a set of new public SLA templates. All these benefits result in satisfied users, who continue to use the market, therefore increasing liquidity in the Cloud market. However, these benefits come with some cost for the user. Whenever a public SLA template has been adapted, the users of this template have to re-define their SLA mappings.

The four contributions of this paper are: (1) the definition of three adaptation methods for adapting public SLA templates to the needs of users; (2) the investigation of conditions under which SLA templates should be adapted; (3) the formalization of measures, i.e., utility and cost, to assess SLA adaptations and SLA adaptation methods; and (4) the introduction of an emulation approach for the defined use cases.

The remainder of the paper is organized as follows: Section 2 describes related work. Section 3 introduces the adaptive SLA mapping approach and the cost–benefit model. The simulation setup, the three adaptation methods, and the simulation infrastructure are described in Section 4. Section 5 presents the simulation results and a discussion. Section 6 concludes the paper.

2. Related work

For putting our work in context of the state-of-the-art, we briefly describe Cloud resource management, Cloud marketplaces, and the existing work on SLA matching.

2.1. Cloud resource management

There is a large body of work about managing resource provisions, negotiations, and federation of Cloud and Grid resources. An example is [8]. They designed agent technology to address the federation problems in Grids, i.e., resource selection and policy reconciliation. [9] propose a new abstraction layer for managing the life cycle of services. It allows automatic service deployment and escalation depending on the service status. This abstraction layer can be positioned on top of different Cloud provider infrastructures, hence mitigating the potential lock-in problem and allowing the transparent federation of Clouds for the execution of services. [10] investigate three novel heuristics for scheduling parallel applications on utility Grids, optimizing the trade-off between time and cost constraints.

However, most of the related work on resource management considers resource provision from the provider's point of view and does not consider Cloud computing infrastructures in the context of a marketplace.

2.2. Cloud market

Currently, a large number of commercial Cloud providers have entered the utility computing market, offering a number of different types of services. These services can be grouped into three types: computing infrastructure services, which are pure computing resources on a pay-per-use basis [11–13]; software services, which are computing resources in combination with a software solution [4,14]; and platform services, which allow customers to create their own services in combination with the help of supporting services of the platform provider. The first type of services, which is also called Infrastructure-as-a-Service (IaaS) consists of a virtual machine, as in the case of Amazon's EC2 service, or in the form of a computing cluster, as done by Tsunamic Technologies. The number of different types of virtual machines offered by a provider is low. For example, Amazon and EMC introduced only three derivations of their basic resource type [3]. Examples for the second type of services, which are called Software-as-a-Service (SaaS) are services offered by Google (Google Apps [4]) and Salesforce.com [14]. These companies provide access to software on pay-per-use basis. These SaaS solutions can hardly be integrated with other solutions, because of their complexity. Examples for the third kind of Cloud services, which are called Platform-as-a-Service (PaaS), are Sun N1 Grid [15], force.com [14], and Microsoft Azure [16]. In this category, the focus lies on provisioning essential basic services that are needed by a large number of applications. These basic services can be ordered on a pay-per-use basis. Although the goal of the PaaS service offerings is a seamless integration with the users' applications, standardization of interfaces is largely absent. Furthermore, big Cloud providers as the mentioned Azure or EC2 do not even provide their SLAs in a standardized format, e.g., XML. If they want to participate in markets with higher liquidity, as leveraged by our approach, they have to comply to the market rules and formalize their SLA templates in a machine-readable way. Nevertheless, the implementation of system resource markets have been discussed in several projects [17–19]. [20] give an overview over information systems for traded resources in Grid markets and [21] deal with economic models of Grid computing markets. All in all, however, mentioned works either do not define the tradable goods, work with very simplified definitions, or do not take market liquidity into account.

2.3. Service level agreement matching

The main SLA matching mechanisms are based on OWL, DAML-S, or similar semantic technologies. [6] describe a framework