PIK, Vol. 34, pp. 80–86  DOI 10.1515/piko.2011.014

# Investigations of an SLA Support System for Cloud Computing (SLACC)

**Guilherme Sperb Machado**

Communication Systems Group CSG
Department of Informatics IFI
University of Zürich
Binzmühlestrasse 14
CH-8050 Zürich
Switzerland
machado@ifi.uzh.ch

*Guilherme Sperb Machado* received his M.Sc. degree from the Federal University of Rio Grande do Sul (UFRGS, Brazil) in 2009, and his B.Sc. degree from the Pontifical Catholic University of Rio Grande do Sul (PUCRS, Brazil) in 2006, both in Computer Science. In 2008, he worked as an intern at HP Labs Bristol, U.K., addressing the area of IT change management. In April 2009 he joined the University of Zürich, Switzerland, Communication Systems Group CSG, where he currently works as a junior research and Ph.D. student, supervised by Prof. Stiller. His Ph.D. topic is in the area of SLA management and estimation of SLA parameters for distributed systems and services, e.g., Cloud Computing. His areas of interest include accounting in distributed systems, Cloud Computing, IT service management, protocol design, network security, and semantic Web in network management.

**Burkhard Stiller**

Communication Systems Group CSG
Department of Informatics IFI
University of Zürich
Binzmühlestrasse 14
CH-8050 Zürich
Switzerland
stiller@ifi.uzh.ch

Prof. Dr. *Burkhard Stiller* chairs as a full professor the Communication Systems Group CSG, Department of Informatics IFI at the University of Zürich UZH since 2004. He holds a Computer Science Diplom and a Ph.D. degree of the University of Karlsruhe, Germany. During his research locations of the Computer Laboratory, University of Cambridge, U.K., the Computer Engineering and Networks Laboratory, ETH Zürich, Switzerland, and the University of Federal Armed Forces, Munich, Germany his main research interests cover, including current CSG topics, charging and accounting of Internet services, economic management, systems with a fully decentralized control (P2P), telecommunication economics, and biometric management systems. He participates in a number of European, industrial, and Swiss research projects, coordinates FP7 SmoothIT and SESERV, and serves as a technical program committee member as well as chair of several conferences.

## Abstract

Cloud Providers (CP) and Cloud Users (CU) need to agree on a set of parameters expressed through Service Level Agreements (SLA) for a given Cloud service. However, even with the existence of many CPs in the market, it is still impossible today to see CPs who guarantee, or at least offer, an SLA specification tailored to CU's interests: not just offering percentage of availability, but also guaranteeing, for example, specific performance parameters for a certain Cloud application. Due to (1) the huge size of CPs' IT infrastructures and (2) the high complexity with multiple inter-dependencies of resources (physical or virtual), the estimation of specific SLA parameters to compose Service Level Objectives (SLOs) with trustful Key Performance Indicators (KPIs) tends to be inaccurate. This paper investigates an SLA Support System for CC (SLACC) which aims to estimate in a formalized methodology – based on available Cloud Computing infrastructure parameters – what CPs will be able to offer/accept as SLOs or KPIs and, as a consequence, which increasing levels of SLA specificity for their customers can be reached.

## 1 Introduction

In the recent past Cloud Computing (CC) received an attention from the ICT (Information and Communication Technology) community due to the conjunction of key aspects, which formed an innovative concept for dynamic provisioning of scalable and virtualized resources over the Internet. Mainly, features like self-service, virtualization, pay-by-use (or pay-on-demand), scalability, high availability, and easy dynamic resource allocation makes CC applicable and a solution for, e.g., processing huge data sets for genetics sequencing and Customer Relationship Management (CRM) services [8].

Within CC environments, a Service Level Agreement (SLA) needs to exist between two parties: Cloud Providers (CP) and Cloud Users (CU), e.g., organizations or individuals. These two parties need to agree on a set of parameters expressed through the SLA. However, even with the existence of many CPs in the market (e.g., Amazon, SalesForce, Rackspace, or Google), it is still impossible today to see CPs, who guarantee or at least offer an SLA specification tailored to CU's interests. However, this is of great importance for tomorrow's CC, since very general requirements (such as the "availability needs of a given service" [13], [1], [18], [17]) do not match commercial needs for guaranteed CC services. Thus, CPs need accurate definitions of objective values. An example of a specific SLA parameter is the Return to Operation (RTO) time, in case of virtual machine failures. If the RTO is estimated, CPs can compose a Service Level Objective (SLO) offering guarantees of Key Performance Indicators

(KPI) with a higher precision (e.g., RTO under 3 minutes, measured by the bootstrap time of virtual machines). Nevertheless, due to (1) the huge size of CPs' IT infrastructures and (2) the high complexity with multiple inter-dependencies of resources (physical or virtual), the estimation of specific SLA parameters to compose SLOs with trustful KPIs tends to be inaccurate. This inaccuracy can result in penalties for a CP, if an unrealistic set of values was proposed. Therefore, the lack of an automated system that maps and aggregates low-level measures into SLOs is the key barrier for (a) less risky and (b) customer-specific SLA-based CC service provisioning.

As far as known today (cf. Section 2), there is no work addressing the problem of mapping low-level measures of interdependent resources into SLOs inherent to typical Cloud services. Moreover, solutions like SLA assessments [10] and SLA monitoring [2] that provide an approach of SLA assessment, do not take into consideration the CC infrastructure as whole, but very specific network parameters only.

Therefore, this paper investigates the SLA Support System for Cloud Computing (SLACC) – decision support system for CC – in order to estimate in a formalized methodology, based on available CC infrastructure parameters, what CPs will be able to offer/accept as SLOs or KPIs and which increasing levels of SLA specificity for their customers can be reached.

The remainder of this paper is organized as follows. Section 2 presents related work and Section 3 describes a related use case. Section 4 defines the set of major requirements for SLACC, while Section 5 introduces relevamt building blocks. Section 6 contains the architecture proposal. Finally, Section 7 provides a summary, also discussing a table for comparing the new approach to related work.

## 2 Related Work

To address the background of the CC area the case of commercially available systems is considered. 25% of large enterprises today – those that have more than 1.000 employees – are already spending money on IaaS (Infrastructure-as-a-Service) via an external CP and those who are not using Cloud services today may consider to use it in a near future [8]. Thus, CC addresses small and medium companies as well as large enterprises. Moreover, another interesting aspect pointed by [8] is that "*the interest in use production applications in the Cloud is nearly as high as for test or development purposes"*. Therefore, commercial demands for CC can be derived, which is stated as a trend on moving important parts of the business into the Cloud.

With the deployment of sensitive services (in terms of business criticality, technical robustness and security, or performance-wise) for end-users and providers, a well-defined and specific SLA is necessary. However, most CPs nowadays just offer general SLA parameters, such as availability. Therefore, Table 1 outlines four large CPs with selected services offered, and for each service listed an overview of its SLA parameters is also presented.

It can be observed that the "availability" appears predominantly in all Cloud SLAs, just with Rackspace offering performance and recovery time guarantees, which determines the only exception. Such a situation is understandable, since assessing other non-trivial parameters may increase the risk of occurring SLA violations and CPs should try to avoid uncertainty. However, offering non-specific SLAs is extremely business critical to CUs. Customers do not demand availability guarantees only, but also the confidence that, e.g., database queries will run in less than *n* seconds on the top of a certain SaaS (Software-as-a-Service) product. This kind of guarantee is a refined SLA parameter, which impacts highly CUs' businesses, especially when CUs use a service in a distributed manner, e.g., a large health insurance company using CC-based CRM in multiple countries, where the rate of querying information about its customers is considerably high. Today, this level of SLA specificity is not offered in the CC.

In the scope of current research, a small number tries to solve problems inherent to SLA management in CC environments. In that respect the SLA@SOI project [6] is focused mainly on dynamic SLA monitoring for diverse distributed systems and provides three main benefits in that CC area:

- *Predictability and Dependability*: Quality characteristics of services can be predicted/enforced at run-time.
- *Transparent SLA Management*: Service level agreements (SLAs) defining the exact conditions under which services are provided/consumed can be transparently managed across the whole business and IT stack.
- *Automation*: The entire process of negotiating SLAs, delivery, and monitoring of services is automated allowing for dynamic/scalable service consumption.

Inside the SLA@SOI context, SLOs – beforehand agreed upon – are constantly monitored, and the system is able to predict at run-time the occurrence of SLA violations. The RESERVOIR project [7] proposes an SLA Protection system, which detects and predicts SLA violations at run-time and takes actions interacting with the Service Lifecycle Manager (SLM). The SLA Protection system monitors SLA parameters to take actions in case of a possible violation (SLA prediction), like reallocating virtual machines or adjusting resources. The SLM deals with low-level components and performs changes in the deployment of Virtual Machines to respect SLA parameters. This SLA Protection system can estimates risks and acts pro-actively to prevent penalties.

The TrustCOM project [12] looked deeply into the subject of SLA negotiation and monitoring, and produced a reference implementation. In the negotiation part, the project does not use an SLA assessment/estimation technique. However, SLA parameters are monitored and a component called SLA Performance Logger accumulates historical data on the performance of SLAs for future evaluation and use.

The AssessGrid project [5] focuses on SLAs and risk management. The architecture brings a Risk Assessment component, which interacts with the monitoring system (having historical data) to check risks of an SLA under negotiation. AssessGrid provides an approach to develop risk values related to existent SLOs and KPIs, e.g., the probability of SLA penalties, if a given SLA is agreed upon. However, AssessGrid does not estimate KPIs in case of lacking knowledge to negotiate SLA parameters, e.g., SLOs and/or KPIs that can be offered. These two distinct approaches are needed: assessing risks of SLOs and KPIs, previously generated by a decision support system as SLACC, diminish possible upcoming

| CP | Service | SLA Parameters |
|---|---|---|
| Amazon [16] | S3 [15] | Availability (99.9%) with the following definitions: Error Rate, Monthly Uptime Percentage, Service Credit |
| | EC2 [13] | Availability (99.95%) with the following definitions: Service Year: 365 days of the year, Annual Percentage Uptime, Region Unavailable/Unavailability, Unavailable: no external connectivity during a five minute period, Eligible Credit Period, Service Credit |
| | SimpleDB [14] | Subject to the Amazon Web Services Customer Agreement, since no specific SLA is defined. Such agreement does not guarantee availability |
| Salesforce [18] | CRM | The company's Web site does not contain information regarding SLAs for this specific service |
| Google [1] | Google Apps (including a.o. GMail business, Google Docs) | Availability (99.9%) with the following definitions: Downtime, Downtime Period: 10 consecutive minutes downtime, Google Apps Covered Services, Monthly Uptime Percentage, Scheduled Downtime, Service, Service Credit |
| Rackspace Cloud [17] | Cloud Server | Availability regarding the following: Internal Network: 100%, Data Center Infrastructure: 100%<br>Performance related to service degradation: Server Migration in case of performance problems: migration is notified 24 hours in advance, and is completed in 3 hours (maximum).<br>Recovery Time: In case of failure, guarantee the restoration/recovery in 1 hour after the problem is identified. |
| | Cloud Sites | Availability: Unplanned Maintenance: 0%, Service Credit |
| | Cloud Files | Availability: 99.9%, Service Credits |

**Table 1** Overview of SLA parameters from large cloud providers.

penalties and enable CPs to be more competitive in a CC negotiation market. Moreover, the need of assessing parameters in SLAs was observed by [2] and [10]. [2] is part of the SLA@SOI project and this approach to assess SLA parameters takes into consideration historical data, i.e., what was accounted for and monitored in a Cloud. For SLA hierarchies the assessment can also consider different levels of contracted SLAs (e.g., SLAs with Internet Providers or all SLAs inherent to the good functionality of a given service). [10] is aware of the fact that most of SLAs parameters are monitored – and kept as historical data – or assessed. However, [10] proposed an approach that relies on the statement that an accurate estimation of network Quality-of-Service (QoS) parameters is not required in most cases: it is sufficient to be aware of service disruptions (i.e., when the QoS provided by the network collapses). [10] proposes an algorithm for a disruption detection of network services. [10] sees an estimation of SLA parameters as critical, it concludes in a higher-level that advantages may be exploitable.

## 3 Use Case

Since SLACC's main objectives include (1) CPs will benefit from SLACC to propose accurate SLA parameters and SLOs/KPIs beforehand and (2) once CPs receive CU requests for dedicated SLOs/KPIs, the CP can evaluate, if such values can be guaranteed in his CC infrastructure, SLACC takes into consideration inter-dependencies of resources inside the CC infrastructure. Thus, the following example describes a use case for a better understanding of the SLACC approach to be proposed. Figure 1 illustrates the use case in a high-level view.

Considering that a CU wants to contract a service with a CP, usually, the most common situation for end-users, is when the CP offers (Figure 1, step 1) a pre-formed – ready for establishment – SLA for specific services (with all SLOs and KPIs determined). Depending on CP, the CU can either accept the pre-formed document or reject it proposing a negotiation phase (Figure 1, step 2). Note that as seen in Section 2, no large commercial CP mentions the possibility to enter in a SLA negotiation phase. Such negotiation tends to be inaccurate (and consequently risky for CPs) due to the huge size of CPs' IT infrastructures and the high complexity with multi inter-dependencies of resources. How the negotiation itself is conducted is out of the scope of SLACC, since the focus is on the estimation of appropriate knowledge and its optimization to the CP to compose valuable and specific SLAs. In other words, SLACC system supports the SLA negotiation process with highly suited values.

During the negotiation, the CU can propose new SLA parameters, e.g., "Minimum Web Service Query Processing Time" (Figure 1, step 3). The SLO for this parameter is "The Query Processing Time related to the Web Service X should be less than 2 seconds". The CP will consult SLACC to know if the proposed SLO is possible with the given values (Figure 1, step 4). Assuming that the SLA and SLO are described in a machine-readable manner (respecting a model) pointing to existent resources in the CC IT infrastructure, SLACC implements an estimation algorithm that infers, looking to Accounting Records databases and the current infrastructure state, if such value (here, 2 seconds) can be satisfied.

After the negotiation phase, the expected result is an SLA that is tailored to CP and CU interests (Figure 1, step 5). It means that CUs may result on contracting the service having

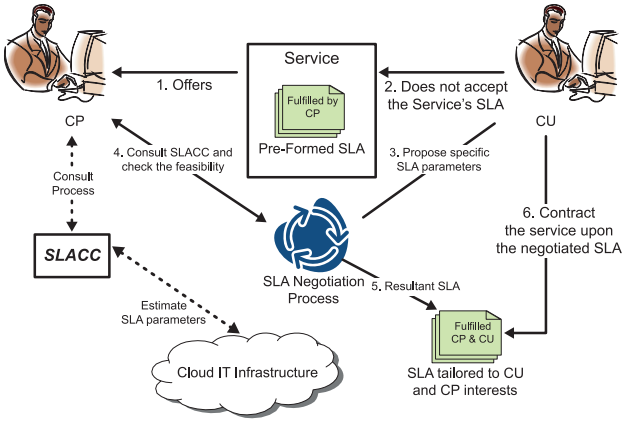


**Figure 1** SLACC solution overview.

a more specific SLA than a pre-formed one (Figure 1, step 6). Therefore, the CP may satisfy customer's needs also having a less risky contract related to penalties.

## 4 SLACC Requirements

Based on the presented use case, the following list of requirements was derived and defines the main functional and performance properties that the SLACC decision support systems needs to comply to:

- The SLACC solution has to be able to *estimate* KPIs having as output a unique value, or a set of values in case that an SLO requires it.
- *Geographical resource* locations and their distribution must be taken into account, since the CC IT infrastructure can be dispersed world-wide. The Cloud Infrastructure Model has to be aware of that, since geographically dispersed resources can impact, for example, the performance estimation.
- The design and implementation of SLACC should be *flexible* in order to work with any KPI based on SLA requirements and specified SLOs. The system should evaluate – by estimating – if, e.g., the bootstrap time of virtual machines (in this case, the KPI) can satisfy the SLO that is set to $n$ minutes.
- The SLACC solution has to be *scalable* in the sense that even within a larger CC IT infrastructure, estimates can be calculated and provided in a plausible amount of time. For example, the SLACC system must not interfere negatively in SLA negotiations due to a possible larger delay in estimates processing.

SLACC needs to offer a *system interface* to the operator, where objectives – related to the estimation – can be adjusted. E.g., the CP may want to estimate the minimum time value for a specific query operation in a CC application, considering using a minimal amount of virtual database instances. Therefore, in this case, the objective related to the estimation will be "the use of less possible resources".

## 5 SLACC Building Blocks

SLACC considers a wider range of parameters inside the CC IT infrastructure, balancing historical information, current IT infrastructure status (e.g., servers load, network bandwidth at the moment), and how the Cloud is organized internally, including all its IT inter-dependencies, e.g., a physical server depends on some switches that are connected at the core network, or virtual machines that have some applications which depends on a set of databases. Thus SLACC building blocks include:

- **Integrated Architecture**: SLACC requires an integrated architecture, where all components combine an end-to-end solution in the scope of CC. These components can interact with existent components by defining clear interfaces between SLACC functionality and CC infrastructures. Such an integrated architecture determines the basis for the next four building blocks.
- **Cloud-specific and Multi-level SLA Model**: Most of the CPs tackle general SLA parameters, such as availability and other service-unspecific performance parameters. Existent SLA languages or models (e.g., SLAng [3], WSLA [11]) should be taken into consideration to design an automated approach to derive SLA parameters for typical CC services inherent to different levels: from IaaS to SaaS levels. The benefit of this approach is that CPs and CUs can define SLA requirements in a higher level of abstraction, while describing SLOs/ KPIs that are typically in the lower level.
- **IT Infrastructure Model**: SLACC has to be based on a formal model reflecting general and typical CC IT infrastructures. Existent approaches, like the Common Information Model (CIM) [4], should be considered and extended to match SLACC's needs. E.g., one extension foreseen today includes the separation of what the CP IT infrastructure is and what the CU infrastructure is, considering virtualization and Operational Systems (OS) that can run multiple other OSs.
- **Algorithms to Estimate SLA Parameter Values**: An estimation algorithm takes as an input a set of SLA parameters from CUs and CPs, and generates as the output a required range of values that the CP will be able to offer – described in terms of SLOs – to the requesting CU. Such an estimation algorithm will be based on estimation theory [9].
- **Estimates Repository**: For practical purposes SLACC must be able to record in a secured and legally compliant manner, which estimations were taken into account at a certain point of time. Thus, an estimates repository needs to be designed to fit commercial and business support system's needs. In turn, it can be used for (a) future estimations, (b) keeping track on how the system acted in the past, providing analytics, and (c) allowing for future fine adjustments in case of lessons learnt.

## 6 Architecture and Estimation Approach

SLACC will estimate SLA parameters, e.g., KPIs based on SLOs, to enable the design of more specific SLA documents.

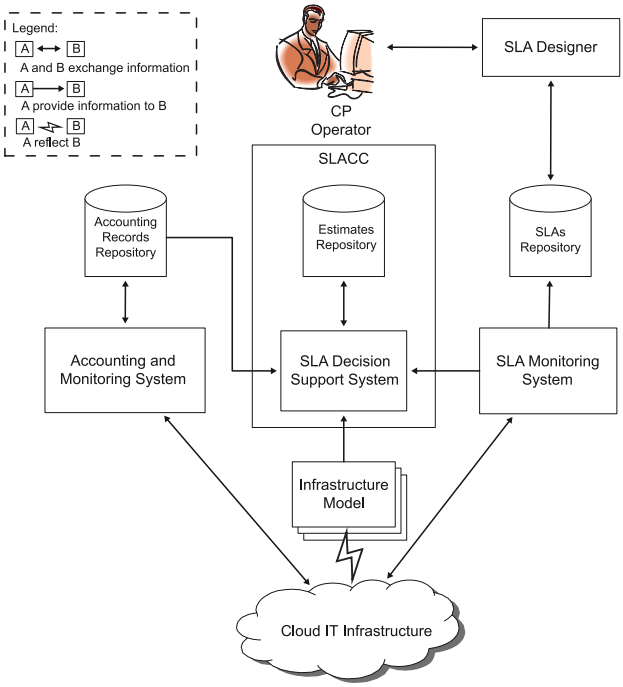


**Figure 2** SLACC architecture.

The system will map high-level requirements into low-level factors that, combined together in a balanced manner, form an estimation. Thus, the following steps are investigated: an integrated architecture, a well-defined Cloud IT Infrastructure Model, and an estimation algorithm.

Figure 2 shows the abstract view of the SLACC architecture, which serves as the starting point for SLACC developments. SLACC interacts with the Accounting Records Repository, the SLA Monitoring System, and the Infrastructure Model. The Infrastructure Model component enables an updated view of all inter-dependencies of the Cloud IT Infrastructure. It is important to reflect exactly the organization of the physical IT environment, otherwise the SLA Decision Support System (DSS) will not estimate SLA parameters in an accurate manner. The Infrastructure Model component provides an updated status information of the managed items inside the Cloud infrastructure (e.g., memory used of a given virtual server). The CP Operator interacts with the SLA Designer in order to build a well-defined SLA, using an SLA model/language. The SLA DSS can be split into sub components: the estimation engine (implementing an estimation algorithm) and others. These sub components' interfaces are defined by an API (Application Programming Interface) to interact with other components of the SLA management architecture. This API will serve as the CPs openness factor and the supporting interface for inter-domain interactions.

The key mechanism within SLACC is the design and development of the algorithm estimating with a defined level of confidence SLA parameters, such as the "minimum database query time" for a given application. The principle operation of the estimation algorithm is as follows. The CU proposes an SLA with a specific SLO, e.g., "RTO of Virtual Machines under 3 minutes". It is known that the Return Time to Operation can be measured in different ways, but the KPI associated to this SLO is measured by a composition of low-level values inherent to the bootstrap of virtual machines.

SLACC consults CC's IT Infrastructure knowledge base (represented by the infrastructure model) to check "what are factors that matter for a successful bootstrap of a virtual machine?". Based on relations defined in the Infrastructure Model, a set of factors are determined. In this case it can be assumed that the following factors were mapped:

- Network bandwidth from the virtual machine's template repository to the physical server, which the virtual machine will be hosted on – assuming a transfer from the repository to the assigned physical server;
- Processing capacity from the physical server, which hosts the virtual server;
- Average workload of the physical server in an interval period of time;
- Time to deploy and configure the specific requested virtual machine template in the virtual server;
- Time to (re)configure the deployed virtual machine in the load balancing front-end of the CP.

The estimation algorithm considers a viable distribution to compose and balance these factors to estimate the final result. The challenge here is to balance different factors like "the processing capacity of a give server" with "the average workload" to come up with a value that can be trusted. At the last step, the CP can evaluate based on known facts, if the SLO "RTO of Virtual Machines under 3 minutes" proposed by the CU can be guaranteed by the CP, or if the CP has to negotiate, in this case, this parameter's value to a higher value, or if the CP has to offer different parameter(s).

In order to evaluate the benefits of SLACC, it must be shown that the system can provide accurate estimates to CPs in order to better enhance its SLAs. Based on the estimates for some SLA parameters, these will be monitored in order to evaluate the confidence level of such generated values. Moreover, to prove the scalability of the proposed solution, comparisons between estimates generated by humans and by SLACC should be taken into consideration.

## 7 Summary and Comparison

This work identifies and describes the problem in CC SLA management, detailed related work, and presents a set of key requirements. Moreover, it proposes the SLACC architecture with a brief discussion on an estimation approach for SLAs, and a system overview and its building blocks. In turn, SLACC will increase the level of SLA specificity, not handling service's availability only, but also a wider range of specific performance parameters.

Taking into consideration related work and its major dimensions of technical functionality, the majority of these dimensions have been collected and applied to a comparison as summarized in Table 2. This table indicates the properties of the SLACC decision support system. As this comparison shows, SLACC composes different features into one system. Therefore, it can be highlighted that the utilization of a IT In-

| Approach | SLA@SOI [6] | RESERVOIR Project [7] | R. Serral-Gracià [10] | Trust COM Project [12] | AssessGrid [5] | SLACC |
|---|---|---|---|---|---|---|
| **Prediction (for)** | Static parameters | Static parameters | Yes, evaluating past service disruptions | No | No | Dynamically added parameters |
| **Range of Parameters** | Narrow | Medium and flexible | Narrow | Wide and flexible | Unknown and flexible | Wide and flexible |
| **Estimation Algorithm** | No | No | No | No | No | Yes |
| **Risk Assessment** | No | No | No | No | Yes | No, embedded into systems |
| **IT Infrastructure Model** | No | Yes, just for virtualization | Unknown | Only partially available | Unknown | Yes |
| **Estimates Repository** | No | No | No | No | Yes, risk repository | Yes |
| **SLA language** | Yes, not defined yet | WS-Agreement | Unknown | WS-Agreement | WS-Agreement | Yes |
| **SLA monitoring** | Yes | Yes | No, detects service disruptions | Yes | Yes | Yes |

**Table 2** Detailed comparison summary.

frastructure model is suited for CC. Furthermore, the estimation algorithm uses balanced infrastructure factors produces relevant results, and the flexibility of adding KPIs and SLA parameters to the system (which can be estimated in a later moment) is backed by an estimate repository.

The respective and general architecture of the SLACC system was described, presenting key components and explaining its roles. Moreover, key aspects concerning the estimation algorithm were discussed, also presenting key differences from other approaches in the area of SLA management.

The upcoming steps for a successful implementation of the SLACC system includes the development of each building block as presented in Section 5. A CC IT Infrastructure Model will be defined and utilized for the implementation as well as the Cloud-specific SLA model. The core of SLACC will be based on such models, which reflect the actual IT infrastructure state and an agreed upon SLA document. Furthermore, it is fundamental to gain a wide knowledge of which parameters inside a CC IT infrastructure matter to ensure an optimal performance in those scenarios and beyond. Thus, it will be possible refine the estimation algorithm balancing several factors in a correct way, without producing values within acceptable thresholds.

## References

[1] Google.com Apps: *Google App Service Level Agreement*. Available at: http://www.google.com/apps/intl/en/terms/sla.html. Last visited on February 2010.

[2] M. Comuzzi, C. Kotsokalis, G. Spanoudakis, R. Yahyapour: *Establishing and Monitoring SLAs in Complex Service Based Systems*, IEEE International Conference on Web Services (ICWS2009), IEEE Computer Society, Washington, DC, USA, 6–10 July 2009, pp 783–790. doi:10.1109/ICWS.2009.47

[3] D.D. Lamanna, J. Skene, W. Emmerich: SLAng: a language for defining service level agreements, The Ninth IEEE Workshop on Future Trends of Distributed Computing Systems, 2003. FTDCS 2003, 28–30 May 2003, Vol. 1, pp 100–106.

[4] Distributed Management Task Force (DMTF) Website: *Common Information Model (CIM)*. Available at: http://www.dmtf.org/standards/cim. Last Visited on May 2010.

[5] J. Padgett, I. Gourlay, K. Djemame (eds), AssessGrid Deliverable 1.3: *System Architecture Specification and Developed Scenarios*, Version 0.30, December 2006.

[6] SLA@SOI Project Website: *Empowering the service industry with SLA-aware infrastructures*. Available at: http://sla-at-soi.eu. Last visited on February 2010.

[7] RESERVOIR Project Website: *Service Manager Scientific Report*. Available at: http://www.reservoir-fp7.eu/fileadmin/reservoir/deliverables/A4_ServiceManager_ScientificReport_V1.0.pdf. Last visited on February 2010.

[8] Forrester Research Website: *Conventional Wisdom is Wrong About Cloud IaaS*. Available at: http://www.forrester.com/rb/Research/conventional_wisdom_is_wrong_about_cloud_iaas/q/id/47102/t/2. Last visited on February 2010.

[9] J. Rice: *Mathematical Statistics and Data Analysis*, Duxbury Press, 2nd Edition, 1st June 1994, ISBN 0-534-209343.

[10] R. Serral-Gracià, Y. Labit, J. Domingo-Pascual, P. Owezarski: *Towards an Efficient Service Level Agreement Assessment*, IEEE Infocom, Rio de Janeiro, Brazil, 19–25 April 2009.

[11] Web Service Level Agreements (WSLA) Project: SLA Compliance Monitoring for e-Business on demand. Available at: http://www.research.ibm.com/wsla. Last visited on February 2010.

[12] The TrustCOM project. Deliverable 64: *Final TrustCoM Reference implementation and associated tools and user manual*, Version 3.0, June 2007.

[13] Amazon.com Web Services: *Amazon Elastic Compute Cloud (EC2) Service Level Agreement.* Available at: http://aws.amazon.com/ec2-sla. Last visited on February 2010.

[14] Amazon.com Web Services: *Amazon SimpleDB*. Available at: http://aws.amazon.com/simpledb. Last visited on February 2010.

[15] Amazon.com Web Services: *Amazon Simple Storage Service (S3) Service Level Agreement.* Available at: http://aws.amazon.com/s3-sla. Last visited on February 2010.

[16] Amazon.com Web Services: *Products and Services*. Available at: http://aws.amazon.com/products. Last visited on February 2010.

[17] RackspaceCloud Website: *RackspaceCloud Service Level Agreements*. Available at: http://www.rackspacecloud.com/legal. Last visited on February 2010.

[18] SalesForce.com Website: *The Leader of Customer Relationship Management (CRM) and Cloud Computing*. Available at: http://www.salesforce.com. Last visited on February 2010.