International Journal of Scientific Research and Management (IJSRM)
||Volume||06||Issue||04||Pages||EC-2018-01-09||2018||
Website: www.ijsrm.in ISSN (e): 2321-3418
Index Copernicus value (2015): 57.47, (2016):93.67, DOI: 10.18535/ijsrm/v6i4.ec01

Service Level Agreement (SLA) for Cloud Computing Compilation with Common and New Formats

Haider Abdul Hassan Hadi Al Kim[1], Shouman Barua[2]

[1]Department of Electronic and Communications Engineering, Faculty of Engineering, University of Kufa, Najaf, Iraq
[2]Faculty of Engineering and Information Technology, University of Technology Sydney, Australia

Abstract:
Future fit demand and flexible solutions combined with high quality of services that is what latest information technology by cloud computing stands for. In the last couple of years, many scientists and researchers have improved computing systems that result in it more users' friendly day by day from all perspectives. Nowadays, a computing system promises to fulfill maximum outcome using minimum efforts or resources from a user's point of view, and it comes from overcoming several technological eras like cluster computing, grid computing, etc. It is obvious that as more the technology advances, the complexity also increases in terms of privacy, security, and service quality. Recently business applications want services being delivered instantly. Pay-as-you-go is the latest trend for most of the business applications and to suit it precisely a process should be followed which might be the Service Level Agreement (SLA). In this paper, we started with defining cloud computing and how it defers from the traditional computing system. We also compile the format and parameters of estimating SLA for cloud computing in order to be benefited maximum from it in terms of users' as well as providers' perspectives.

Keywords: Cloud Computing; Service Level Agreements; Utility Computing; Saas; Iaas.

1. Introduction

To refer an analogy about cloud computing, a popular question is always asked. What would be preferred if someone needs milk? There are two options. 1) Buying a cow 2) buying a bottle of milk. The answer to this question is not straightforward. In order to take a decision on this question, many factors should be considered. Business is a continuous process and highly dynamic, especially in the technological environment. To cope with the latest trend, and in order to meet the revenue target, a cost optimization (CAPEX, OPEX) process is attempted very often. Using cloud computer (Compared with buying bottled milk) turns the business into a new era where several parties are involved, and they are aiming for a common goal. Since this system involves more than one party, the agreement should be formatted, where all the terms and condition are stated. Apparently, the format defers with every business aspect. Could computing is the latest trend. Therefore, the format is in compiling the agreement is also a new concept, which is updating day by day.

With the advancement of the technology and business standard, Computing is becoming a fundamental need like other existing utilities such as electricity, water, telephone, gas , etc. [4]. It is being altered into the model involving services that are delivered and commoditized in a similar way to traditional services aforementioned before. Many paradigms have already been built up to meet the vision of this utility computing. Cluster computing and Grid computing are prominent paradigms followed by cloud computing, the latest technology for a utility computing system.

Cloud computing requires a clear agreed SLA signed by the service consumer and committed by service provider like other utility services mentioned earlier. Coming up with an agreement is one of the most important parts of the cloud computing. It is unlike other utility agreement, and the process of this agreement is not mature enough. Many researchers are trying to define a perfect SLA for cloud computing, but things are not such easy since billions of money do the matter in this field. To define about SLA, the simplest definition is that it is a contract negotiated and agreed between the vendor who provides the service and the user who consumes the service.

In the second part of this paper, cloud computing and the basic differences between cloud computing and cluster computing as well as grid computing are reviewed. The third part is exclusively for SLA that is one of the key factors for operating cloud computing. The reminder parts of this paper are organized specifications of cloud computing SLA like how it works, languages, different SLA formats, differences, similarities and results of the discussion. Finally, we conclude the paper with some recommendations for our findings.

2. Cloud Computing: New Paradigm of Utility Computing.

To define cluster, grid, and cloud, various definitions were attempted from a number of researchers and practitioners.

According to the Pfister [6] and Buyya [7] cluster computing is as follows:

"A cluster is a type of parallel and distributed system, which consists of a collection of inter-connected stand-alone computers working together as a single integrated computing resource."

Buyya defined one of the popular definitions for Grids at the 2002 Grid Planet Conference, San Jose, USA as follows:

"A Grid is a type of parallel and distributed system that enables the sharing, selection, and aggregation of geographically distributed 'autonomous' resources dynamically at runtime depending on their availability, capability, performance, cost, and users' quality-of-service requirements."

Buyya also defined [5] cloud computing as follows:

"A Cloud is a type of parallel and distributed system consisting of a collection of inter-connected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resource(s) based on service-level agreements established through negotiation between the service provider and consumers."

Scalability, pay-per-use utility model, and virtualizations are the key factors for the cloud computing. Cloud computing is listed as below according to the National Standards and Technology (NIST):

• On-demand access to the network so-called convenient.

• Computing capacity of a shared cluster of configurable computers.

• A network that is released and been available rapidly with less effort in management.

• Cloud model comprises

1. Five essential properties (self-services provided on demand, wide access to a network, combining of resources, accelerated resilience, calculated services)

2. Four deployment clouds' models are name: Public and Private Clouds, as well as Community and Hybrid Clouds)

3. Three service models (Service represented by Platform i.e., Google App Engine, Service represented by Infrastructure i.e., Amazon EC2/S3, and Service represented by Software i.e., salesforce.com)

The structure of the total cloud computing process is as follows [1]:

Users or Brokers just contact with the utility computing system through applications to submit a request in the first level (Fig 1). Admission Control (AC) is conducted at the second level by Service Request Examiner (SRE). Then, Resource Allocation is performed by Management layer below the (SRE). Finally, resources or services are allocated by Service Provider or Resource.

Each layer in below layered-design might be considered as a single or group of parties who are engaged in the calculation, e-business, and outsourcing process to delimit the lower commitments and anticipations that agreed between contracted groups. A perfect SLA must involve specifications in both general and technical sides. This includes pricing policies, business groups, and the required resources' properties to handle the service [1].

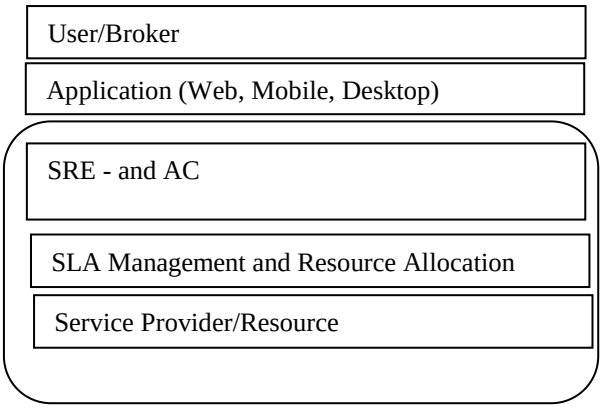

Figure 1: Basic structure of Cloud computing system.

3. Service Level Agreement (SLA)

SLA can be defined from various points of view like web services, networking, the internet, data management center, etc. Though they vary from field to field, however, it is true that overall functions of all the service level agreements are almost same and so true for the cloud computing as well.

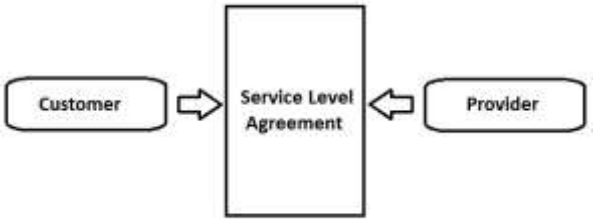

Figure 2: Position of the Service Level Agreement.

Fig 2 shows a block diagram how SLA is positioned. It is a contract between the service provider and service consumer. Service customers and service providers must abide by the rules that are specified in the agreement. In this agreement, the business relationship between these two parties is clearly focused.

Dinesh et. al. [1] defined SLA as: "An explicit statement of expectations and obligations that exist in a business relationship between two organizations: the service provider and customer".

In order to form a good SLA, following facts should be considered.

• A service should be described extensively in the format.

• Service performance (QoS) should be presented.

• It must define the mechanisms by which the determinants of provided service can be supervised.

• Penalties must be imposed in case of failure to fulfill the service requirements. The rate of penalty should also be there.

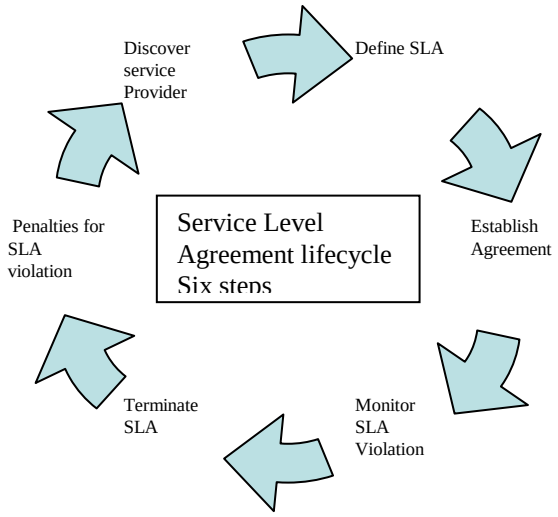

Figure 3: SLA life cycle characterized by Sun Microsystems Internet Data Center Group.

According to Jin et al. [1] a SLA should have several components like aim, limitations, domain, period of validity, Service Level Objective (SLO), contacting sides, sanctions and administration

4. SLA Validation and Management

This section of the paper is divided into:

4.1 SLA Validation framework:

SLA acts as a safeguard for the service consumers. It always helps a customer to have a safe mode because providers always tend to be flexible about their services. In e-Business platforms, every time unit might count thousands even millions of revenue. So SLA should be validated accordingly, and there should have strong management to make an agreement and to take a decision about any incident. A SLA validation framework [2] might be based on the LAYSI infrastructure model [8], LoM2HiS management model [9], and SLA aggregation and validation model [10].

LoM2HiS (Low-Level Metrics to High Level Service Level Agreement) is a Resource-Level validation system which supplies a mean to convert metrics of resource to parameters placed in high-level service. It offers resource provision that is automatically manageable, in turn, it will synchronize resource provision in low-level with SLO. The service provider uses this system to maintain the quality of services that are mentioned in SLA [2]. It is an integral component of the FoSII project (Foundations of Self-governing ICT Infrastructures)

LAYSI is an Infrastructure-Level validation, which is an approach that is based on a layered

structure to prevent the violation of SLA [2]. It allows self-adaptable, scalable, and exchangeable components. It is also included in FoSII project (Foundations of Self-governing IXR Infrastructures). It has a workflow service that is responsible for finding out a possibility of threat violation during time response. It escalates the violation threat to the Automatic Service Deployment Layer (ASD) if it fails to fix or define the issue. ASD layer explores its history to find out if there is a similar situation comparing with previous experiments. In this way, it prevents the violation of Service Level Agreement.
Privacy, trust, and security are also important factors to be considered during SLA contracts. Because of the details of the aggregation are prevented at each layer, a good strategy called "a distributed top-down validation mechanism" is utilized for the whole validation of a hierarchical SLA aggregation.

4.2 Service Level Agreement Management framework:

Place To use in business application, we must require some management teams at various level whose will be responsible for completing workflow and matters involved with this. Following concepts regarding management framework, were developed in [2].

The Service Level Agreement Manager is responsible for setting the agreement, preparing templates and other instances through the registry. In addition to this, they also take part during the negotiation of SLA period and later after, they monitor the functions.

The Service Manager is responsible for monitoring the necessary elements for service initiation.

The Service Evaluation Manager just predicts the non-functional behaviors of a whole assigned composed service. When the SLA is known, and all dependencies to the lower level are solved, it is called fully specified [3].

Business Manager is on top able to maintain a relationship between a service customer and a service provider.

The architecture can be very flexible and always dynamic from every business's perspective. Even all the matters mentioned above could be controlled by a single SLA Manager and a single Service Manager respectively. Fig 4 shows an overview of manager architecture. For extensive specifications, we refer [2].

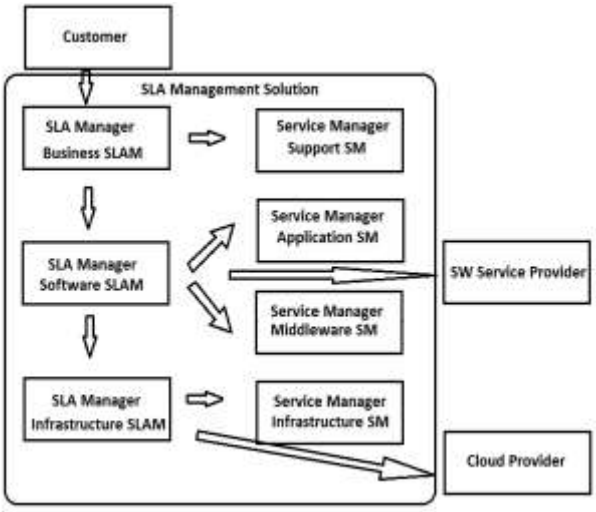

Figure 4: SLA Management architecture for business application [2].

5. SLA Management Languages

Special programming languages are used to represent a Service Level Agreement. Negotiation parties are gathered with each other by protocols and language of negotiation s, which decide the way that they are engaged by each other. Various languages have been observed like Bilateral Protocol, WS-Agreement, QML Web Service Offerings Language, SLAng, Web Service Level Agreement, Quality Objects, etc. We will try to give some short descriptions of them and among them, we will describe WS-Agreement and WSLA extensively whose are used widely in research and industry. Providers often offer their services using some templates. A customer chooses services based on templates provided by service provider, and they use a particular template to start a particular negotiation. Template based negotiation is implemented by one or more of those languages' frameworks. A comparison of SLA management languages is given in the table (Table 1).

WS-Agreement: Open Grid Forum (OGF) [11] defined standardization for employing SLA in environments that are distributed service-oriented. This protocol is based on Extended Markup Language (XML). This standard is for the specification and creation of Service Level Agreements, which is widely known as Web Services Agreement Specifications (WS-Agreement). It enables service's users dynamically creating SLA with service providers in order to acquiring services. In addition to this, it also specifies the essential techniques to look after the condition of an agreement as well as to assess the guarantees whose are related an agreement. Moreover, the agreement creation over a template

mechanism is supported by this protocol language. Usually, service provider offers services in a template format and service consumers follow the template to do an agreement.

The template may vary in terms of alternating the option to attract services consumers according to their requirements. This approach might be compared to a supermarket where customers choose the desired produce from the variety of products according to their need. Overviews of the WS-Agreement negotiation components are depicted [11] in Fig 5 where Negotiation Factory creates the Negotiation Process that implements the Negotiation Factory Port Type. Negotiation instance represents the negotiation process, which performs the volitional Advertisement Port Type and the essential Negotiation Port Type.

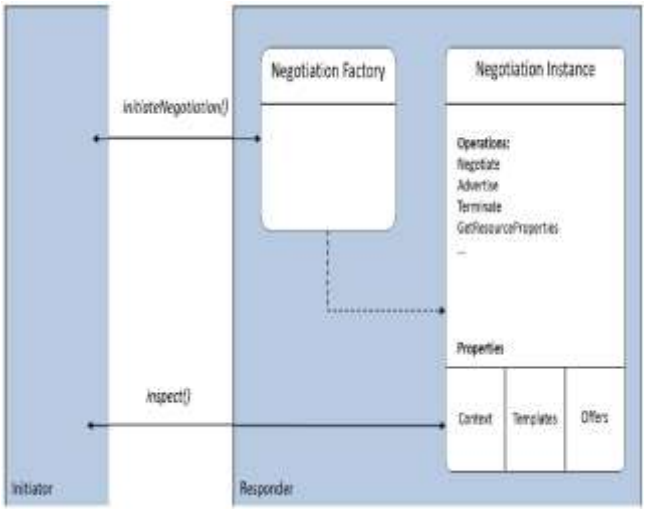

Figure 5: Overviews of the WS-Agreement negotiation components [11].

Web Service Level Agreement (WSLA): To specify and monitor SLA for services WSLA is a language framework developed by IBM [12]. WSLA defines the obligation between service providers and service consumers. It is based on formal XML language to carry Service Level Agreements and architecture to interpret this language at runtime. In the case of SLA's violation, it works on the guarantees of the agreed service requirements to be met, for instance, notifies the customer about the provided service regularly. However, only agreed common view of a service between service providers and service consumers is covered by the WSLA. Typically, it is necessary to translate the obligation of a WSLA into configurable system-level information, which could be owned by any involved party. Both service providers and service consumers can use the WSLA to configure their own system to provide and supervise their services [12]. One of the important aspects of WSLA is its ability to handle with specifics of particular domain and technologies.

Web Service Offering Languages (WSOL): Web Service Offering Language is a XML-based language, which is compatible with Web Service Description Language (WSDL). WSOL describes some functionality whose are not formally described by WSDL. It is usable for monitoring and management specification to reduce the runtime overhead [13]. WSOL is a respectable language used for classes services when the formal specification is needed; management statement's various constraints for web services. According to the [13], a comparison between WSDL and WSOL is depicted in Fig 6.

SLAng: It is also based on XML language for defining Service Level Agreements. Service Level Agreements language (SLAng) [15] is considered different from other languages from three different aspects [14].

Firstly, while other languages focus on web services exclusively, SLAng explains SLA term for a wide range of internet services. Secondly, the SLAng structure is designed based on the requirement of industry [9]. Thirdly, it is officially specified in terms of behaviors of users and services included in the usage of the service. However, for commercial computing it is not that much helpful because, it is not able to define management information.

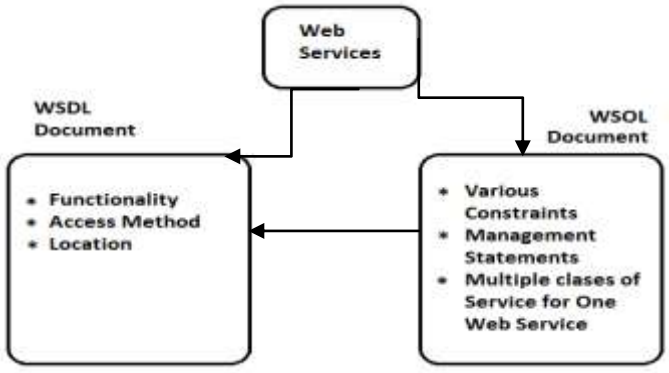

Figure 6: Comparison between WSDL and WSOL [13].

QML: To define the multi-category Quality of Service (QoS) specifications for components, Quality of Service Modeling Language (QML) was presented by Hewlett-Packard [16]. QML might be used to describe the Quality of Services (QoS) properties of a software component, but its specification cannot be executed to implement the

specified QoS. With the help of QML, users can define dimension types of their own. QoS specifications in QML expedite the static corruption of system software into elements with accurately defined QoS limits.

QuO: Quality object is a Common Object Request Broker Architecture (CORBA) specified framework to provide a quality of service in network-centric distributed applications [17]. QoS parameters, notifications, and adaptations can be described by using QuO, which includes a quality descriptions' language. QuO integrates and simplifies information from many providers, locations, and times to help the developers. The new version of QuO 3.1 supports CORBA 3, CIAO, MICO, Redhat Linux 9, and macosx. It also works with both Java and C++ applications.Bilateral Protocol: it is a good kind of negotiation mechanism for the advance resource reservation presented by Srikumar Buyya [1]. It is based on Rubinsteins Alternating offers protocol for negotiating Service Level Agreements between parties. It can be modified by any party with the objective of reaching a mutually agreed contract. Bilateral Protocol has been implemented by Srikumar Buyya and Linlin Wu using the Grid Bus Broker on the customer's part and Aneka on the provider's side [1]. This service is platform independent i.e. users and providers can use individual platform.

Table 1: Basic comparison of SLA management languages

Name	*Based on*	*Domain*	*Metrics*	*Presented by*	*SLA Lifecycle*
WS-Agreement	XML	Any domain	Have restrictions	Open Grid Forum	Step 1 to 6
WSLA	XML	Originally for Web services	New metrics allowed	IBM	Step 1 to 6
WSOL	XML	Originally for Web services	N/A	V Tosic, K Patel, B Pagurek [13]	Step 1 to 4
SLAng	XML	Originally for environment of internet DS	No	Lamanna, Skene, Emmerich [15]	Step 1 to 4
QML	Not specified	Any domain	New metrics allowed	HP	Step 1 to 4
QuO	CORBA specified framework	Any domain	N/A	CORBA	Step 1 to 4
Bilateral Protocol	.Net, Java,	Originally for resource reservation	Yes	Srikumar Buyya	Step 1 to 4

6. Cloud Computing Parameters and Standardization for SLA

Before going into the details on this, one might ask, why cloud or why not any other option is to consider, what are the factors for a business to drive it towards the cloud computing solution? To some extents, the answer is already cleared. In addition to this, one might choose cloud computing because of its operational excellence, improve the valued customers' demand, cost optimization, and speed up the production by reducing time. A SLA template must include all aspects of SLA components. It starts playing a role from the very begging for the services start. Especially it plays a very important role when any violation is observed. Therefore, SLA should be well defined. The most common metrics for preparing a SLA template are as follows:

Uptime: It is the time how long a system or a machine is running uninterruptedly. Reliability and availability can be measured by this easily.

Average duration of outage: If we take the total summed time for all outage and divide it by the total number of occurrences, we get the average amount of time that the system is down.

Unplanned outage time: this is an unexpected service unavailability, which is not desired at all from a quality service provider. In this period, the system goes down due to system failure.

Downtime: It specifies the whole period of time that the system is unavailable. In other word, it is the system failure of a certain period. We can calculate the reliability, availability from this metric.

MTBF: It defines the expected amount of time that the service is unavailable during an operation.

In addition to these, some parameters are specially specified for individual services:

1) Parameters for Infrastructure as a Service (IaaS)

[5] are availability, Storage, CPU capacity, Memory size, Boot time, Scale uptime, Scale downtime.

2) Parameters for Platform as a Service (PaaS) are Pay-as-you-go billing (charging based on time of service or resources), Browsers, Scalability (a degree of freedom to use with a large number of online users), and servers.

3) Parameters for Software as a Service (SaaS) are availability: how long time software is available to use, scalability: using with an individual or a large organization, customizability: flexible to use with different types of users.

Apart from this, geographical location (location of data stored), scalability (ability to increase or decrease storage space), security (authorization, cryptography for storage, authentication), privacy, backup, recovery (in case of disaster or failure), transferring bandwidth (maximum speed for transferring data) should be clearly specified in a SLA template.

We recommend a SLA should have the following parts:

a) Document Control and Stakeholders
c) Service Management and Service Scope
e) Primary contacts and Support Hours
f) Customer Requirements and IT organization requirements
i) Infrastructure Services and Service Transactions
k) Reporting, Changes & Review
m) Agreement termination, Approval & Signature

7. Other recommendations Typical Values For Some Parameters

To have an impression, it is possible to set some values, which are typical to choose a cloud computing service. SLA should also belong to those values with the parameter metrics. For instance, a network uptime target should not be less than 99.99% especially for North America, Australia and Europe regions and 99.95% for the rest of the regions. A server uptime target should also be at least 99.99% for North America, Australia and Europe regions and 99.95 for the rest of the world. Response time target should not be more than 3 minutes for an emergency incident. A latency target must be considered less than one millisecond. Other metrics are set according to the business requirement, and a perfect negotiation is required in order to complete the SLA between two parties.

VIII. OBSTACLE TO PREPARE SLA FOR CLOUD COMPUTING

Cloud computing is not a much matured technology yet, and it is facing still a lot of problems which is to be solved. Extensive researches are ongoing to fix the bugs. There might have some solutions for those problems, but they are not always convenient. Some major problems are scalable storage, performance unpredictability, data transfer bottlenecks, data lock-in, software licensing, scaling quickly, Bugs in large scale distributed systems, data confidentiality and audit ability, availability of services, etc. problems to be solved

It is already tried to give an idea how a perfect SLA might look like. However, there is no doubt that success of cloud computing still depends on some trustful relationship between service consumers and providers. Privacy and security are two big issues in a cloud computing system. There are not adequate legal instruments or tools to facilitate cloud computing to be trustful. Local law is also a barrier to implement cloud computing all over the world because it is an on-demand service offered to everywhere in the world at a time. Suppose we consider a SLA between the United States of American provider and an Australian consumer. This is an open question that what law enforcement should be followed to prepare a perfect agreement.

If data is damaged or lost, the questions are: how the parties will deal the issues? What are the consequences? There are limited specifications for this. It needs more research to improve this incident.

If it is needed to terminate the service level agreement, consumers have to face an immense trouble which is not convenient to go into the could computing system. Once decided, it is essentially needed to remove all the associated configuration information from the service system. It is also a debatable issue who the parties are, who take care of this activity and what the impacts are on it? It is expected to come up with some recommendations for those problems in our next extended paper.

8. Wrap Up

Despite of progressing huge, some questions are yet unsolved. Service is yet to use efficiently. As stated earlier several parties are involved with a cloud computing system. Therefore, it is difficult to manage quality service provided by different parties at a time. If any party fails to fulfill the agreement, total workflow will be affected. Developing an integral methodology to have a pre threat with sufficient response time before the violation occurred could solve the issues. However, it is yet to

develop properly for all the parameters stated in the SLA.

In this paper, it is tried to review the concept of cloud computing along with service level specification, which is the integral part of cloud computing workflow. The metrics and parameters are tried to accommodate for a service template. It also focused some issues, which are still unsolved and still hot as a research topic. Finally, it might be said that a perfect SLA specification that is mentioned in this paper could be a perfect template for agreeing with a business relationship between service consumer and service provider.

By solving all of the above problems, a cloud-computing system would be definitely a second to none choice for all the consumers presented in the market. However, it will require more time and extensive research.

9. References

[1] Wu, L., & Buyya, R. (2010). Service level agreement (SLA) in utility computing systems. arXiv preprint arXiv:1010.2881.

[2] Haq, I. U., Brandic, I., & Schikuta, E. (2010). Sla validation in layered cloud infrastructures. In Economics of Grids, Clouds, Systems, and Services (pp. 153-164). Springer Berlin Heidelberg.

[3] Theilmann, W., Winkler, U., Happe, J., & de Abril, I. M. (2010). Managing on-demand business applications with hierarchical service level agreements. In Future Internet-FIS 2010 (pp. 97-106). Springer Berlin Heidelberg.

[4] Buyya, R., Yeo, C. S., Venugopal, S., Broberg, J., & Brandic, I. (2009). Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. Future Generation computer systems, 25(6), 599-616.

[5] Alhamad, M., Dillon, T., & Chang, E. (2010, April). Conceptual SLA framework for cloud computing. In Digital Ecosystems and Technologies (DEST), 2010 4th IEEE International Conference on (pp. 606-610). IEEE..

[6] Gregory Pfister, "In Search of Clusters", 2nd Edition, Prentice Hall, USA, 1998.

[7] Buyya, R. (1999). High Performance Cluster Computing: Architecture and Systems, Volume I. Prentice Hall, Upper SaddleRiver, NJ, USA, 1, 999.

[8] Brandic, I., Emeakaroha, V. C., Maurer, M., Dustdar, S., Acs, S., Kertesz, A., & Kecskemeti, G. (2010, July). Laysi: A layered approach for sla-violation propagation in self-manageable cloud infrastructures. In Computer Software and Applications Conference Workshops (COMPSACW), 2010 IEEE 34th Annual (pp. 365-370). IEEE.

[9] Emeakaroha, V. C., Brandic, I., Maurer, M., & Dustdar, S. (2010, June). Low level metrics to high level SLAs-LoM2HiS framework: Bridging the gap between monitored metrics and SLA parameters in cloud environments. In High Performance Computing and Simulation (HPCS), 2010 International Conference on (pp. 48-54). IEEE.

[10] Haq, I. U., Paschke, A., Schikuta, E., & Boley, H. (2009, May). Rule-based workflow validation of hierarchical service level agreements. In Grid and Pervasive Computing Conference, 2009. GPC'09. Workshops at the (pp. 96-103). IEEE.

[11] Waeldrich, O., Battré, D., Brazier, F., Clark, K., Oey, M., Papaspyrou, A.,. & Ziegler, W. (2011). Ws-agreement negotiation version 1.0. In Open Grid Forum.[Accessed 20 May 2013]. Available from: https://www.ogf.org/documents/GFD.193.pdf

[12] Ludwig, H., Keller, A., Dan, A., King, R. P., & Franck, R. (2003). Web service level agreement (WSLA) language specification. IBM Corporation, 815-824.

[13] Tosic, V., Patel, K., & Pagurek, B. (2002). Wsol—web service offerings language. In Web Services, E-Business, and the Semantic Web (pp. 57-67). Springer Berlin Heidelberg

[14] Skene, J., Lamanna, D. D., & Emmerich, W. (2004, May). Precise service level agreements. In Proceedings of the 26th International Conference on Software Engineering (pp. 179-188). IEEE Computer Society.

[15] Lamanna, D. D., Skene, J., & Emmerich, W. (2003). SLAng: a language for service level agreements. Technical Report D2, University College London, March 2003

[16] Frølund, S., & Koistinen, J. (1998). Qml: A language for quality of service specification. Hewlett-Packard Laboratories.

[17] Intelligent Distributed Computing Department. Distributed Systems Technology Group Technologies home page. "Quality of Object (QuO ", Available at http://www.dist-systems.bbn.com/tech/QuO, January 2006

Author Profile

Haider Al Kim got the B.Sc. in Information and Communication Engineering from Al-khwarizmi Engineering College, University of Baghdad , Baghdad, Iraq in 2008. Master degree in Telecommunication Networks from University of Technology Sydney (UTS), Sydney, Australia in 2014 under the supervision A. Prof. Kumbesan Sandrasegaran. Working and research areas are Wireless Telecommunication, Mobile Network, Network Management, Network Design and Implementation ,Data Analysis and Monitoring , OOP with C++ , Broadband and Communication Protocols. He is a Cisco instructor at Al Mansour College, Baghdad, Iraq. He hold Cisco Certificate in 2010 with Cisco ID CSCO11773718. Alcatel-Lucent SAM certification holder, Alcatel University, Sydney Australia 2013. From 2015-2018 He is working as assistant lecturer at University of Kufa in ECE department – Faculty of Engineering. Into PhD at TUM University - Germany (Starts in October 2018).

Shouman Barua is a PhD research scholar at the University of Technology, Sydney. He received his BSc in Electrical and Electronic Engineering from Chittagong University of Engineering and Technology, Bangladesh and MSc in Information and Communication Engineering from Technische Universität Darmstadt (Technical University of Darmstadt), Germany in 2006 and 2014 respectively. He holds also more than five years extensive working experience in telecommunication sector in various roles including network planning and operation.