Toward Cloud Computing QoS Architecture: Analysis of Cloud Systems and Cloud Services

M. H. Ghahramani, *Member, IEEE,* MengChu Zhou, *Fellow, IEEE,* and Chi Tin Hon, *Senior Member, IEEE*

***Abstract*—Cloud can be defined as a new computing paradigm that provides scalable, on-demand, and virtualized resources for users. In this style of computing, users can access a shared pool of computing resources which are provisioned with minimal management efforts of users. Yet there are some obstacles and concerns about the use of clouds. Guaranteeing quality of service (QoS) by service providers can be regarded as one of the main concerns for companies tending to use it. Service provisioning in clouds is based on service level agreements representing a contract negotiated between users and providers. According to this contract, if a provider cannot satisfy its agreed application requirements, it should pay penalties as compensation. In this paper, we intend to carry out a comprehensive survey on the models proposed in literature with respect to the implementation principles to address the QoS guarantee issue.**

***Index Terms*—Cloud computing, quality of service (QoS), service level agreements (SLA), system-of-systems (SoS).**

I. Introduction

CLOUD provides scalable, on-demand, and virtualized resources for users. Users can use a shared pool of computing resources provisioned with their minimal management efforts [1]. Moreover, resources are provided by the pay-per-use model and users just pay for resources they use. Avoiding over-provisioning and under-provisioning of resources and reducing the cost of deploying hardware can be considered as some motivations for companies to take their business into the clouds [2]. Various service models, namely software as a service (SaaS), platform as a service (PaaS), and infrastructure as a service (IaaS) are proposed for clouds. Moreover, private, community, public, and hybrid clouds can be regarded as deployment models for clouds. Users can choose one of these services and deployment models in terms of their requirements. Apart from the aforementioned advantages of using clouds, they face some obstacles and concerns [3]. One of such concerns is how to have quality of service (QoS) guaranteed by service providers. User applications with various characteristics and demands are executed in clouds. To define requirements of each application, users and cloud service providers negotiate a contract called service level agreement (SLA). According to it, if a provider fails to meet the mutually agreed SLA, the provider is penalized. Designing architectures and algorithms for managing SLAs in clouds is still in its early stages and in fact, there are profound challenges for enhancing traditional algorithms to satisfy SLAs properly. In [4], [5] authors present surveys on QoS in clouds but more parameters should be scrutinized in more specific details to provide a comprehensive vision. Existing definitions of clouds QoS lack a comprehensible taxonomy that can be easy to understand. Thus, in this paper we intend to provide a deep insight into QoS issues by reviewing the technical details of QoS metrics and classifying them. By designing a comprehensive taxonomy, we determine what QoS scenarios exist and what their characteristics are. The remainder of this paper is organized as follows: quality of service (QoS) in clouds is reviewed in Section II; the resource management is examined in Section III; Section IV scrutinizes monitoring platforms to evaluate the QoS in clouds; the challenges and conclusions are given in Sections V and VI, respectively.

II. Quality of Service in Cloud Computing

The evolution of cloud has its roots in cluster computing, utility computing, and grid. QoS has been a major topic of interest within grid — a system that coordinates resources subject to no centralised control. The need to effectively manage large-scale resource-intensive scientific applications across multiple administrative domains was the research motivation in that area. Cloud also shares this motivation within a new context oriented towards business needs, for reliable service stipulation rather than just resource provisioning for scientific batch-oriented applications. Although web services in grid emerged as a paradigm of sharing resources for collaboration and resource usage optimization purposes, these two aforementioned paradigms are different in the aspects related to their programming/business models, software dependencies, management, usability, processing performance and security.

This section aims to introduce a general taxonomy of QoS in order to provide a complete overview of the problem. Defining the appropriate QoS metrics is an important aspect of this problem, because of its direct relationship with service level agreement. Cloud computing systems usually host many clients and users at any given time. The latter may

Manuscript received July 23, 2016; accepted October 28, 2016. This work was supported by Fundo para o Desenvolvimento das Ciencias e da Tecnologia (FDCT) (119/2014/A3). Recommended by Associate Editor Long Chen.

Citation: M. H. Ghahramani, M. C. Zhou, and C. T. Hon, "Toward cloud computing QoS architecture: analysis of cloud systems and cloud services," *IEEE/CAA Journal of Automatica Sinica*, vol. 4, no. 1, pp. 6−18, Jan. 2017.

M. H. Ghahramani is with the Institute of Systems Engineering, Macau University of Science and Technology, Macau 999078, China (e-mail: sphr.ghahramani@gmail.com).

M. C. Zhou is with the Institute of Systems Engineering, Macau University of Science and Technology, Macau 999078, China, and also with the Helen and John C. Hartmann Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ 07102, USA (e-mail: zhou@njit.edu).

C. T. Hon is with the School of Management, Fudan University, Shanghai 200433, China (e-mail: cthon@ieee.org).

Digital Object Identifier 10.1109/JAS.2017.7510313

access different types of services and resources with varying requirements. In order to meet their requirements and perform the services, it is necessary to provide a certain level of QoS by service providers. Providing a guaranteed QoS in a widely distributed platform hosting services is not an easy task [6], [7]. Researchers have developed mechanisms, frameworks, and systems to guarantee the QoS requirements of different services. This section takes an in-depth look at them.

A. Service Level Agreement (SLA) Issues

As we mentioned before, the relationship between a cloud provider and user must be described with an SLA. Because users trust providers to deliver some of the latter's cloud services, it is vital to define those services, how they are delivered and how they are used [8]. A contract is a formal agreement between two or more parties to create mutual relations or legal obligations. Contracts consist of different parts, such as the definition of business partners, specification of functional obligations, and quality, price, and penalties related to the participating parties [9]. Although managing and monitoring the quality levels of services rely on automated tools, nowadays the cloud SLAs are typically plain-text documents, and sometimes the service providers publish them online. The examples are the Amazon S3 SLA [10], and Amazon EC2 SLA [11]. They provide a facility to agree upon QoS between an end-user and provider and define end-user resource requirements and provider guarantees, thus assuring end-users that they are receiving the services they have paid for [12]. In [13] the authors consider two broad phases for managing a cloud service from a user's perspective, the pre-interaction phase and post-interaction phase. In the first phase users want to select a cloud service provider to initiate a service for the first time. In the latter, while already using a service, they want to monitor the performance of their selected service, as well as the other available services, to assess whether or not the provider continues to provide the same QoS as it was at the time of service selection and to consider service migration if another service, that offers the same or better QoS at a lower cost becomes available.

Interoperability among clouds remains a challenging issue. Therefore, intercloud scenario emerges as a solution to define new cloud architecture by creating cloud of clouds. To address this challenge a cloud service broker (CSB) is considered as a third party providing interoperability where it consolidates services from multiple providers, and provides a uniform interface. An advantage of using CSB is that it simplifies service selection which indirectly eliminates vendor lock-in for users [14]. Marrouf *et al.* introduce a trusted third party (TTP) based on multi-agent systems (MASs) in order to control the QoS contract and guarantee transparency and symmetry with respect to the SLA contract between prospective signatories [15]. This TTP uses MASs to apply an advanced penalty model that guarantees the performance and reliability of the cloud. In [16] Motta *et al.* have presented a third-party SLM (service level management) framework that covers the whole service life cycle that meets the typical requirements of SLM in clouds. As the strong dialectics between customers and providers, it overcomes the limits of popular frameworks, e.g., information technology infrastructure library (ITIL) and control objectives for information and related technology (COBIT). The proposed SLM, by covering the SLM lifecycle, guides users to follow a simple approach.

B. SLAs Mechanisms

SLAs act as the main method to assure the quality level of cloud services. However, there is no standard for the implementation of SLAs developed for cloud models. In order to design and implement SLAs, the following aspects should be considered. First, one needs a clear SLA structure based on a predefined ontology of cloud. Second, the proposed SLAs should be linked to the QoS metrics and the cost model to provide an acceptable framework for users. Finally, reliable monitoring tools are required for testing metrics related to SLAs. Several frameworks such as web service level agreement (WSLA) [17], web services agreement (WS-agreement) [18], and service level agreement language (SLAng) [19] are proposed for SLA management. The work [9] contributes to the research on web service matchmaking and ranking in two ways. First it proposes an extensible QoS model for web services that are suitable to manage different QoS dimensions, either negotiable, non-negotiable, domain dependent, or domain independent, defined by heterogeneous metrics and measurement methods. Second, it proposes mechanisms for service matchmaking and selection, and offers configuration able to exploit the aforementioned QoS description. However, several challenges such as automatic negotiation and dynamic SLA management according to the environmental changes remain to be addressed and their resolution needs more investigations. In [20], Huang describes a service-oriented cloud computing platform that enables web-delivery of application-based services with a set of common business and operational services. In addition, various resource monitoring systems are necessary for monitoring QoS metrics when applications are running in clouds. Due to the dynamic nature of clouds, the matching of SLA templates needs to be dynamic and continuous QoS monitoring is necessary to enforce SLAs. An SLA template contains many parameters like cloud resources such as physical memory, main memory, processor speed and properties like availability and response time [21]. Hence monitoring tools are essential in ascertaining cloud activities like resource planning, resource management, and performance management. They are essential in providing fault tolerance and the migration of tasks in the event of a resource failure. Monitoring can be of two types: high-level and low-level. High-level monitoring is related to the virtual platform status. The low-level monitoring is related to information collected for the status of the physical infrastructure. In the following we identify such capabilities as availability, elasticity, timeliness, resilience, and reliability that an ideal monitoring tool should possess to serve the objectives.

C. QoS Metrics

As mentioned earlier, cloud monitoring is needed for constant measurements to appraise resources in clouds in

terms of performance management, resource management, and ability to meet SLAs. Monitoring tests can be divided into computation-based test and network-based one [22]. The former is related to the status of real or virtualized platforms running cloud applications. Data metrics considered in such test include CPU utilization, CPU speed, disk throughput, VM acquisition, and system up-time. The latter is focused on network-layer metrics like bandwidth, jitter, round-trip time, throughput, traffic volume, and service response time [23]. Both tests should be applied in order to detect, or even anticipate system behavior changes that could have an important impact on QoS.

At runtime, a set of operations takes place in order to meet the QoS metrics specified in SLA document that guarantees the required performance objectives of users. Being aware of the system's current software and hardware service status is imperative for handling uncertainties to ensure the fulfillment of QoS objectives. In addition, detecting failures while deploying software services on hardware resources is essential. Uncertainties can be tackled through the development of efficient, scalable monitoring tools [24]. Fig. 1 presents the classification of our selected QoS metrics.

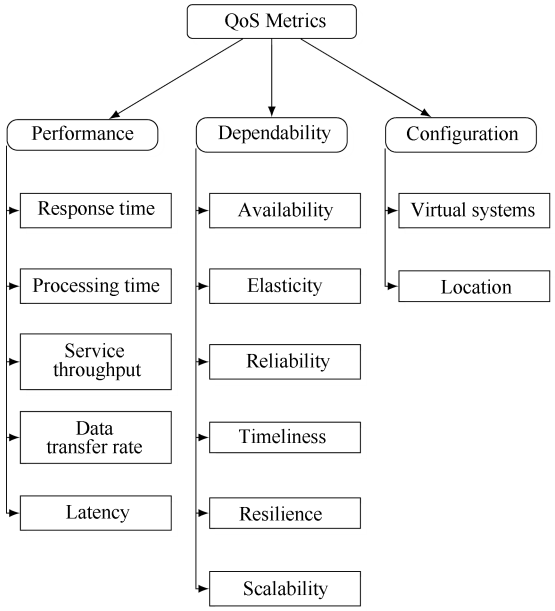

Fig. 1. Classification of QoS metrics.

They are classified into performance, dependability, and configuration. Performance indicates how well a service performs. It is divided into metrics time, ratio, and latency. Time includes various metrics such as response time (R_T) and processing time. The former is defined as the time elapsed from a user's submitting a service request until the final output is received. The latter refers to average response time.

$$R_T = T_f - T_s \tag{1}$$

where T_f is the time instant a user receives the needed output from a provider and T_s is the time the user sends a task request to the provider.

Ratio indicates performance metrics which are computed based on ratio, e.g., service throughput and data transfer rate. Throughput is the number of tasks completed by the cloud service per unit of time. It depends on several factors that can affect the execution of a task. Data transfer rate measures how fast the service is provided. Latency is a difference between the moment when a first packet bit passed the input checkpoint and that when the last packet bit passed the output checkpoint at the other end of a channel. It depends on several parameters, e.g., packet length, network latency, and throughput of a provider's network.

Dependability includes availability, reliability, elasticity, timeliness, resilience, scalability, and security. Availability defines the percentage of time when a resource is ready for immediate use and customers can access it. It is given by:

$$\text{Availibility} = \frac{T - T_N}{T} \tag{2}$$

where T is the concerned time period, and T_N is the total time for which the resource is down thus not available. Availability can also be expressed in terms of average downtime per week, month or year or as total downtime for a given week, month or year.

The resources are distributed over servers or sub-systems and each sub-system can work independently. The transactions would be distributed over all sub-systems providing load balancing. Each system component is broken down into its parts and the availability of each part is reported separately, e.g., firewall and databases.

Reliability refers to ability to ensure a continuous process of the program without loss and is characterized by the number of failures, mean time to failure promised by the provider which is the length of time a device or an application is expected to last in operation, and mean time to recovery. It is measured by:

$$\begin{aligned} \text{Reliability} &= P_r[V] \times P[\overline{T}_F] \\ &= \left(1 - \frac{n_f}{n}\right) P[\overline{T}_F] \end{aligned} \tag{3}$$

where $P_r[V]$ is the probability of violation, n_f is the number of users who have experienced a failure, n is the number of users in a concerned time period, and $P[\overline{T}_F]$ is the promised mean time to failure [25].

Elasticity refers to clouds' ability to expand and contract overtime in response to user demands. It is the power to scale computing resources up and down easily when responding to those demands. Scalability is important in determining whether a system can handle a large number of application requests. It has two dimensions. Horizontal scalability (S_h) which means increasing cloud resources of the same types and vertical scalability (S_v) that is defined as the ability to increase the capacity of a cloud [26]. The former can be measured as the increase in resources by initiating more services (e.g., VMs) of the same type during peak load. The latter can be calculated as the maximum available increase in the resources of a cloud. Three states can be considered as follows: 1) Just-in-need state which denotes a balanced state and the workload can be properly handled. In this state, user-demanded QoS can be satisfactorily guaranteed. 2) Under-provisioning state occurs when the requested resources exceed the available allocated resources. It delays the processing of workload and may be at

the risk of breaking QoS commitment. 3) Over-provisioning state occurs when the allocated resources exceed the requested ones. Although in this state QoS can be achieved, but it leads to additional costs. Different bounds of the hypothetical states can be considered. Let T_m be the total measuring time including all the periods in the states above:

$$T_m = T_j + T_u + T_o \tag{4}$$

where T_u is the under-provisioning time that the cloud platform needs to switch from an under-provisioning state to a corresponding balanced state and T_o denotes the over-provisioning time in which cloud platform needs to switch from an over-provisioning state to a balanced one in those periods of time [27].

Cloud elasticity E_c is the percentage of time when the platform is in just-in-need states; that is:

$$E_c = \frac{T_j}{T_m} = 1 - \frac{T_u + T_o}{T_m}. \tag{5}$$

In [28], the authors have introduced a metric for coverage of scalability to measure the average amount of allocated resources over that of requested resources:

$$\text{Coverage of scalability} = \frac{\sum_{i=1}^{m} \frac{Ar(i)}{R_r(i)}}{m} \tag{6}$$

where m is the total number of requests to extend resources, $Ar(i)$ is the amount of allocated resources of the i-th request, and $R_r(i)$ is the total amount of requested resources of the i-th request. The range of this metric is 0 to 1. The value 1 implies that all requested resources are allocated. Although this formula reflects the ability of the cloud to fit the increase of the workload, but it lacks the description of the level to which the cloud can deliver the same level of performance when the number of requests increases. Scalability models have been proposed in [29]−[31]. The main difference between them is in the selection of the performance metrics used to characterize the system's behavior. The performance quantification is a challenging task due to a variety of metrics, e.g., power, efficiency, and speedup. There are other challenges related to the definition of such scalability metric. First, no a priori knowledge about the QoS requirements and user application is supposed. Second, the enhancement of different components, i.e., CPU, memory, time, and cost should be considered in a kind of the same unit. Third, in order to avoid overheads and costs, scalability measurement has to be kept simple. Finally, the proposed metric should be based on the measurement of parameters that are available through SLA [32].

Timeliness is an ability with which a cloud monitoring system is able to supply information in time when users need to access it. It measures the total time taken to complete a task. This is measured as delay, or latency, or time to complete. This metric also measures the start time and deadline for a task. Total completion time can be given by:

$$T_L(i) = T_{\text{wait}}(i) + T_{\text{exec}}(i) + T_{\text{trans}}(i) \tag{7}$$

where $T_L(i)$ represents the total completion time, $T_{\text{wait}}(i)$ is the waiting time, $T_{\text{exec}}(i)$ is the task execution time and $T_{\text{trans}}(i)$ indicates the transmission time of the i-th task during VM migration [33].

The delay time T_{Dij} of user i to access resources on cloud j is determined by:

$$T_{Dij} = T_{Rij} + T_{Drr} + T_{Nij} \tag{8}$$

where T_{Rij} is the response time from the resources, T_{Drr} is the delay time in resource redirection caused by the redirection overhead for the purpose of load balancing, and T_{Nij} is the network time the request takes to travel from client i to cloud j [29].

Resilience is the ability of a system exposed to hazards to resist and recover in a timely and efficient manner, including the preservation and restoration of its essential basic structures and functions [34]. A resilient system is one that shows reduced failure probabilities, reduced consequences from failures, and reduced time to recovery. It is supported in cloud architectures by employing redundant connections and by duplicating resources or information. Resilience by itself is characterised by several metrics, such as availability and security. Mechanisms aiming to maximize it must consider these criteria as well as common network metrics, i.e., path capacity, one way delay and packet loss [35]. Security metrics deal specifically with policies and mechanisms related to data security that need to be provided to the applications. We can define two security parameters, level of confidentiality and level of integrity.

Configuration includes several metrics indicating the configuration status. It is the process of setting hardware and software details for components of a cloud to ensure that they can interoperate and communicate. These metrics differ among the three main components of cloud provision. In the SaaS distribution model, applications are hosted by a provider, usually over the internet. Configuration can be enabled for the users so that they can make the same types of changes to customize a locally-hosted application. In the PaaS model, operating systems and associated services are delivered over the internet without installation. IaaS involves outsourcing the equipment used to support operations, including storage, hardware, servers and networking components. Virtual systems identify the cloud virtual infrastructure. The location metrics are based on QoS location affinity.

III. Resource Management

Resource management in distributed systems refers to the efficient allocation of workload to shared computing resources. The workload modeling involves the assessment of the arrival rates of user requests and demand for resources, and QoS observed in response to such demands. Due to on-demand characteristics of cloud services, resource management has a tendency to be intrinsically dynamic. For dynamic resource allocation solutions based on SLA, the prediction of workload is one of the important steps. Indeed, although future user behaviors and interactions are largely unknown, the system must remain operational. Many efforts have been made to address the problem of SLA-driven resource management. Some of them consider probabilistic SLA constraints with violation

penalty, while others propose utility function approaches. Different resources, i.e., CPU, memory, and bandwidth can be configured for VMs through a scheduling policy. The existing application's scheduling strategies in clouds are based on the approaches developed in other related areas of distributed systems such as grids. Nabrzyski *et al.* define resource management as the process of identifying requirements, matching resources to applications, allocating those resources, and scheduling and monitoring Grid resources over time in order to run grid applications as efficiently as possible [36]. Grid applications compete for resources that are very different in nature, including processors, data, scientific instruments, networks, and other services. This situation is being complicated by the general lack of available data about the current system and the competing needs of users, resource owners, and administrators of the system. Cao *et al.* have proposed an optimized algorithm for task scheduling based on activity based costing [37]. They have investigated the cost of scheduling different applications and the overhead the applications cause in resource allocation. Their approach considers cost as the only SLA objective for scheduling tasks in a cloud environment. Pandey *et al.* [38] have discussed an optimization-based heuristic for scheduling applications in clouds. They have concentrated on minimizing the total execution cost of applications on cloud resources thereby achieving low computational cost and low data transmission cost. In another work, a temporal task scheduling algorithm is proposed to dynamically schedule arriving tasks to clouds. The cost minimization problem is modeled as a mixed integer linear program and a hybrid simulated-annealing particle swarm optimization is applied to solve it [39]. In order to guarantee the demands to be met, providers have to quantify capacity and resources (e.g., CPU, memory, and storage), and determine the estimated workload. Previous resource management systems such as Condor [40], Globus toolkit [41], load sharing facility (LSF) [42], and portable batch system (PBS) [43] are system-centric mechanisms. In Globus, customers describe their required resources through a resource specification language that is based on a predefined schema of a resource database. The task of mapping specifications to actual resources is performed by a resource co-allocator, which is responsible for coordinating the allocation and management of resources at multiple sites. In fact, they are designed to maximize the overall cluster performance and system utility. The main dimension along which workload management frameworks can be classified is whether the overall objective under which they operate is customer or system-centric. The objective of the former is to maximize the utility of individual customers [44], [45] and that of the latter is to maximize a system level utility, such as fair sharing of resources, minimizing total system cost, or maximizing systems resource utilization [46], [47].

A. Load Balancing

Apart from high resource utilization, various SLA objectives as well as load balanced provisioning of resources should be considered in the service request scheduling [48]−[51]. Several studies have surveyed the characteristics of the QoS showed by the deployment environments of clouds, e.g., VM startup time, and network bandwidth [52], [53]. In [54] the authors have implemented a model to predict and capture workloads in cloud environments for efficient provision of cloud resources. To evaluate the performance of a cloud system the QoS metrics such as response time and reliability should be considered. Resource management and load balancing apply optimal control techniques to manage cloud infrastructure. In [55], Feng *et al.* have proposed an optimal resource allocation model for revenue maximization. For balancing CPU load, Tang *et al.* [56] propose a heuristic solution for VM placement over a very large number of servers in an IaaS data center. Virtualization provides servers with dynamic load balancing. Furthermore, VM migration is proposed to improve the functionality of clouds. It is suggested to remap the overloaded VMs with available resources which have enough processing power to complete task execution. Chen *et al.* propose an optimal load balancing mechanism called EuQoS for scheduling VMs [57]. By extending it to accommodate real-time services, they have implemented the Hadoop platform into the EuQoS system. To investigate the performance of system throughput they have utilized the log processing services. In [58] the authors propose a load balancing technique by using the endocrine algorithm as inspired from regulation behavior of a human hormone system. It achieves system load balancing by applying a self-organizing method among overloaded VMs. Their technique is structured based on communications among VMs. It helps the overloaded VMs transfer extra tasks to another under-loaded VM by applying particle swarm optimization (PSO). Dasgupta *et al.* [59] have proposed a genetic algorithm (GA) over a load balancing strategy to achieve high resource utilization in cloud environment. To simulate the proposed load balancing strategy a cloud simulator is used. Additionally, some existing approaches like first come first serve, round robin, and a local search algorithm called stochastic hill climbing are used in simulation. GA utilizes all the dedicated resources associated with it. It also supports the QoS requirement.

Load balancing algorithms can be categorized into two main groups. According to which phase workload is assigned at compilation or execution time, they can be static or dynamic. Since it is not possible to completely predict task arrival patterns in cloud and due to difference in node capacities, controlling workload to improve performance and maintain stability is crucially important. Consequently, dynamic load balancing approaches tend to provide better performance in such dynamic and distributed environments. Tasks can move dynamically from an overloaded node to an under-loaded one and the performance in terms of scalability and efficient resource utilization would be guaranteed. Although static approaches are more stable and simpler to implement and bring low additional costs, they are not well suited for heterogeneous environment. As balancing techniques ensure the efficient use of physical infrastructure, by modeling the workload dependencies among VMs and distributing workloads in a performance-aware manner, their use can improve the revenue by minimizing, for example, the rejected tasks and maximizing the quality of services. Table I compares some of load balancing algorithms in terms of their pros and cons.

TABLE I
COMPARISON OF LOAD BALANCING ALGORITHMS

Algorithm	Pros	Cons	Environment
First come first serve [60]	Simple to implement.	It is non-preemptive.	Static
Random allocation [60]	Simple to implement.	Overloading may occur.	Static
Round robin [61]	Better than random allocation due to equal workload distribution.	Job processing time is not considered.	Static
Throttled [62]	Distributing the load among the VMs and accessing that load easily; high fault tolerance.	Processing time is not considered for each individual request.	Static
Genetic algorithm [22]	High response time; Using in heterogeneous environment.	High complexity and computational time.	Dynamic
Heuristic based load balanced scheduling [63]	High response time and fault tolerance.	Inherit the first come first serve algorithm.	Dynamic
VM-assign load balancing [64]	Allocating incoming requests to available VMs efficiently.	Does not fit in heterogeneous cloud environment and is not fault tolerant.	Static
Particle swarm optimization [63]	High response time; Useful in heterogeneous environment.	High algorithm complexity and communication overhead.	Dynamic
Min-min [60], [65]	Reducing the makespan.	Existing load on a resource is not considered, thereby leading to starvation.	Static
Max-min [60], [65]	Reducing the makespan.	Smaller jobs may have to wait for long time.	Static
Opportunistic load balancing [66]	Keeping each node in the cloud busy.	Does not consider the present workload of the VMs.	Static
Load balance min-min [67]	Effective task assignment to different nodes, and avoiding unnecessary duplicated assignment.	Job priorities are not considered and response time is low.	Dynamic

B. Resource Allocation

Resource allocation is one of the challenges of clouds because users should access resources from anywhere and expect the latter to be available in any time. Extensively, it is desirable that resource allocation in a cloud environment can be performed dynamically and automatically, based on a fair price and on client high-level requirements. For this to occur, it is important to understand the computational resources and their quantity that should be allocated to a client in clouds specifically. In [68] Batista *et al.* present performance evaluation that considers different resource configurations in a cloud environment to define which dimension of resource scaling has a real impact on client applications. Rajeshwari *et al.* [69] have designed a framework to balance the load among the servers and offer QoS in clouds. They have proposed a two-stage scheduling algorithm. The servers with different processing power are grouped into different clusters. In the first stage, an SLA-based scheduling algorithm determines the priority of the tasks and assigns them to the respective clusters. Thereafter in the second stage, an idle-server monitoring algorithm balances the load among the servers within each cluster.

Resource allocation is one of the most important challenges in distributed systems especially when the clients have SLAs and the total profit in the system depends on how the system can meet these SLAs sufficiently [70]. Elasticity has now become the fundamental feature of clouds as it reflects the ability to automatically add or remove VM instances when workload changes. More specifically, when the workload of a service increases rapidly, an idle framework should respond to the growing performance requirement efficiently. In [71], Chen *et al.* propose a framework for reconfiguration optimization and VM deployment. They have considered an optimization method for resource allocation to maximize the resource utilization and have reduced the cost of runtime reconfiguration by considering the balance of multidimensional resource utilization. In [72] Liu *et al.* introduce a resource management framework to ensure high-level QoS in clouds. It utilizes an aggressive resource provisioning strategy to substantially increase the resources in each adaptation cycle when workload increases. Wu *et al.* [73] have proposed algorithms for resource allocation for SaaS providers to balance the SLA violations and cost of hardware. They have considered such QoS parameters as response time to satisfy users. In order to minimize the costs, they have proposed to reuse the created VMs, which may, however, create adverse security consequences.

A market-based framework for the allocation of different resource requests in clouds has been presented by Fujiwara *et al.* [74]. The resources are delivered and virtualized to end-users as a different set of services. Meanwhile, their approach allows end-users to request an arbitrary combination of services from different providers. Their mechanism utilizes the forward and spot market independently to make flexible and predictable allocations at the same time. Moreover, their proposed technique considers the massive amount of provider ownership incentives for cooperation. Hence, their algorithm maximizes computing as a utility from the service providers. In [75] the authors have introduced a prediction approach to identify the cloud available zone that maximizes satisfaction of an incoming request against a set of requirements. Their

prediction model is built from historical usage data for each available zone and is updated as the features of the zones and requests change. Then, it employs machine learning techniques to learn the unpublished attributes of available zones. The model is dynamically updated to reflect the most recent performance changes in available zones. But the authors did not investigate the confidence level of the prediction models and its influence on overall decision making results, in terms of user satisfaction and cloud performance. Moreover, user requirements are dynamically changing, thus, managing and assigning all available resources in a timely manner is imperative.

To deal with service coordination, i.e., selection, composition, negotiation, and monitoring, an automated QoS-aware approach should be studied and adopted. Therefore, resource recommendation methods that can regulate multi-attribute matching between consumer demands and provider solutions should be designed. By considering QoS metrics, a resource matching algorithm is described in [76]. The authors propose a resource recommendation method for cloud systems, which integrates multi-attribute matching metric, customer evaluation, and price. As consumers and providers are the two main parts in QoS management, the service selection objective is based on roles between them. Besides, a method to provide support for reaching agreements between them is crucial. However, reaching an agreement is even more challenging in the case of service composition where a consumer negotiates with multiple providers. Bi *et al.* present an application-aware approach based on SLA, to optimize the profit of virtualized cloud data centers by proposing an external/internal request arrival rate model for VMs at different service classes [77]. Due to the coexistence of different types of services, their composition has become more complex in a network and cloud environment [78]. Huang *et al.* propose a QoS-aware service approach in order to achieve network-cloud service composition with end-to-end performance guarantee. To do so, they first present a system model for network-cloud service composition and then formulate the service composition problem. Then, by proposing an approximation algorithm and analyzing its properties, they have tried to show its efficiency for QoS-aware network-cloud service composition [79].

Although key players in the IaaS market like Amazon EC2 and GoGrid among others constitute various deployment models by using virtual frameworks, none of them provides a solution for composing services based on users' functional and non-functional requirements, i.e., cost, reliability and latency constraints. Our review concludes that although a large number of methods are available for managing resources, the approaches taking heterogeneity of applications into account are limited and should be explored in the future. In a nutshell, an efficient resource management method should perform QoS-aware utilization of resources, while minimizing the total cost and energy usage. Table II summarizes the strengths and weaknesses of works done by various researchers.

TABLE II
SUMMARY OF STRENGTHS AND WEAKNESSES OF PROPOSED METHODS AND FRAMEWORKS

Ref.	Strengths	Weaknesses
[68]	Handling different configurations of VMs.	Unable to control the elasticity.
[69]	Satisfying SLA as well as load balancing among the servers.	Lacking the implementation detail in IaaS, PaaS, and SaaS.
[71]	Able to improve the profit of platform providers.	Does not consider the parameters that can negatively impact maximizing resource utilization.
[73]	Able to minimize SLA violations.	No clear explanation on an SLA negotiation process.
[74]	Able to maximize computing as a utility from service providers.	Lacks the payment management.
[75]	Able to identify cloud availability to maximize the satisfaction of an incoming request.	Does not investigate the confidence level of the prediction model.
[72]	Increasing resource utilization when workload increases.	Does not investigate multiple interactions and interoperability parameter.

IV. EVALUATING QOS

Monitoring tools are essential in ascertaining the availability of resources and providing feedback to schedulers. They enable guarantees to be made on the performance of any given resource by making sure that computational resources are accessible but not over utilized. Also they are essential in providing fault tolerance and the migration of tasks in the event of a resource failure. With monitoring techniques, cloud providers can acquire information about available resources, application performance status, and predict and detect SLA violations. To address this issue, there is a need to integrate knowledge management techniques into cloud management infrastructure. The monitoring techniques should be capable of automatically determining optimal measurement intervals for different applications, be minimally intrusive to the system, and support large-scale clouds. Various monitoring systems such as GridEye [80], NetLogger [81], Sandpiper [82], Beside storage resource broker, and integrated rule-oriented data system (iRODS) are proposed for grid. Nonetheless, using such monitoring systems in clouds needs some changes and enforcing these algorithms in a cloud environment is associated with some difficulties. In fact, since virtualization is used in clouds, resources are more abstract than those in grids. Therefore, monitoring and measuring QoS metrics become more difficult in this situation. Traditional approaches fall short in addressing these challenges because they examine QoS from a limited perspective. In [83] Hershey *et al.* present a system of systems (SoS) approach to enable QoS monitoring, management, and response for enterprise systems that deliver computing as a service through a cloud computing environment. They have introduced a SoS to provide a clear and concise view of QoS change events within cloud environments. The system can proactively inform enterprise operators of the state of the enterprise, thereby, enabling a timely operator response to any QoS problems. Cloud owners require management and monitoring tools to ensure the cloud performance, robustness, and dependability. In [84] Ma *et al.* have proposed a framework to perform end-to-end measurements at VM instances and

software in the public clouds to monitor QoS parameters of the IaaS and SaaS layers without modifying the implementation of the monitored object. They then discuss the manager-agent and module centralized architecture.

A. Monitoring Systems

Several monitoring systems are proposed in the literature. Each one has its own characteristics and particular monitoring utilities. In the following we classify the available mechanisms and analyze their pros and cons. Nagios [85] is an open source monitoring platform, offering complete monitoring and alerting for cloud infrastructures. It can monitor the status of resources and in this case it checks the hosts and services by using several external plugins and return the status information to administrative contacts. Although it includes valuable features and abilities, it does not support a generic API and performs under small-time interval. In [85] Yeh *et al.* have introduced a monitoring system called SIAM, which is based on Nagios. SIAM supports full monitoring services for SRB/iRODS-based systems, including fault-tolerance and notification functions. Their study focuses on extending existing components and notification functions to satisfy client needs.

Ganglia [86] is a distributed monitoring system for high-performance clusters and grids, which has a hierarchical architecture and relies on a multicast-based listen/announce protocol to monitor the states of systems within clusters and uses a tree of point-to-point connections among representative cluster nodes. Iosup *et al.* [87] analyze the performance of many-task applications on clouds. Correspondingly, many performance monitoring and analysis tools are also proposed. By utilizing these tools they can use the data to rank and measure the QoS of various cloud services according to user applications. Clayman *et al.* [88] establish a monitoring framework for clouds, which enjoys high scalability of monitoring. It spreads in different layers, e.g., service, virtual environment, and physical resources. It offers libraries and tools to build a monitoring system. However, they do not present performance metrics and point out the boundaries. Although many solutions are now available, cloud monitoring technology has not kept pace, partially owing to the lack of open source solutions [89].

Well-known clouds in industry all have their own monitoring utilities. These major enterprises like Amazon, Microsoft and Google offer their own monitoring services, i.e., Amazon web services (AWS) [10], [90], [91], Microsoft Azure [92], Google App Engine [93], and GoGrid [94]. AWS offers three centralized services that enable monitoring and QoS management of applications hosted on its computing and storage cloud services. These services include CloudWatch [10] for monitoring. It is able to monitor services like EC2, in which collected information is mainly related to the virtual platforms. Elastic load balancer [91] is for load-balancing and auto scaler [89] for automatic application scaling and descaling. AzureWatch monitors and aggregates performance metrics from the Azure resources, e.g., instances, databases, storage, and applications. Azure fabric controller [91] is a service that monitors, maintains and provisions CPU services to host the applications that the developer creates and deploys in the Microsoft Azure cloud. CloudStatus [74], which monitors user application performance, is a methodology for analyzing the root cause of abrupt performance changes and degradations, and provides both real time and weekly trends of monitored metrics. The main advertised feature of such platform is its timeliness. VMware vRealize Hyperic [95] is a component of VMware vRealize operations. It monitors operating systems, middleware and applications running in physical, virtual and cloud environments. Nimsoft monitoring solution [96] is able to monitor data centers of both private and public clouds. It has been used for monitoring the status of fulfilling SLAs. Maintaining service levels and availability requires to monitor and manage critical infrastructure in cloud environments. Fig. 2 shows CA Nimsoft monitoring solutions include monitoring environments in public or private cloud. GoGrid offers a centralized load-balancing service called F5 load balancer [93] for distributing application service traffic across servers. Table III shows some of the properties of monitoring tools.

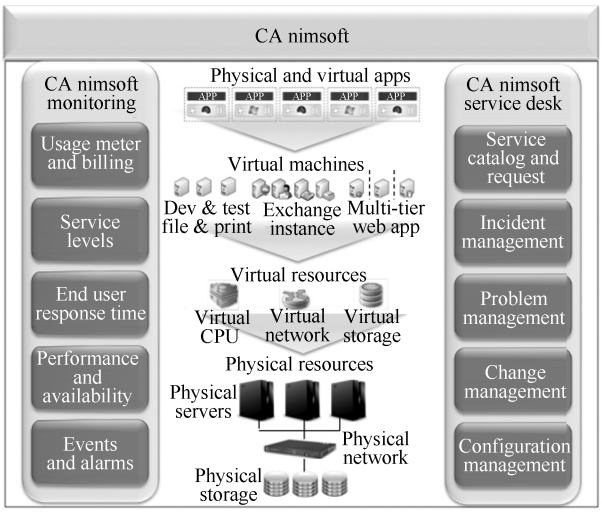

Fig. 2. Nimsoft unified monitoring system.

Along with the increasing role of cloud to the service industry, monitoring as well as the applications deployed are becoming a priority. In [97] Ostermann *et al.* have presented an evaluation of the usefulness of the cloud computing services for scientific computing and the performance of the Amazon EC2 platform have been analysed by using kernels and micro-benchmarks. Similarly a comprehensive performance evaluation of a large computing cloud have been performed. In [98] the authors scrutinize the performance of migration-enabled and error-prone clouds. They propose some stochastic performance modeling and analysis tools.

As an open-source framework for clouds, Eucalyptus implements the IaaS. It is compatible with EC2 and S3 Amazon services to achieve the ability to run and control entire VM instances deployed across a variety of physical resources. Eucalyptus [99] implements a hierarchical network architecture for monitoring the status of CPU, storage, and network services. Since a VM is a key component of cloud computing technology, developing an optimal scheduling mechanism for balancing VM operations in clouds is a critical issue

TABLE III
PROPERTIES OF CLOUD MONITORING TOOLS

Tools	Scalability	Elasticity	Timeliness	Interoperability	Resilience	Availability	Portability
CloudWatch	✓	✓	✓				
AzureWatch	✓		✓				
CloudStatus			✓			✓	
vRealize Hyperic				✓		✓	
Nimsoft	✓			✓	✓		✓
Monitis				✓			✓
Aneka	✓	✓					
CloudKick		✓					✓
Up.time			✓		✓		

for cloud performance.

There is a large number of solutions for monitoring cloud applications. As we mentioned in this section each of these is focused on certain specific aspects of cloud operation, namely availability, elasticity, timeliness, resilience, and reliability, covering only a partial solution for the cloud monitoring problem. In consequence, providing all features of cloud monitoring would require a combination of several monitoring tools, leading to system overhead. Cloud monitoring systems need to be advanced and customized to meet the scalability requirements, and to adapt to highly dynamic environments.

B. Performance Evaluation

In line with our discussion in the prior sections, monitoring tools can be categorized into two types: generic purpose tools and commercial ones. The former are not designed specifically to fit for full cloud operational management. They tend to lack some capabilities like metrics verification. To be used in a dynamic cloud environment, they need to be redesigned and developed. For instance, Nagios and Ganglia have not intended to be used for monitoring virtual resources. They have been designed for slightly changing resources and did not consider elasticity as a requirement to monitor distributed systems. On the other hand, some tools for grids already existed for a resource monitoring purpose and have been developed for monitoring clouds. Such capabilities are essential for efficient management. Redesigning these tools/services should aim to reduce the complexity of computing administration by addressing low-level and high-level cloud-specific monitoring issues. The commercial tools are oriented to optimize performance and resource availability but lack portability and interoperability due to the proprietary issue. These tools are generally designed for monitoring commercial platform-dependent clouds. Take Amazon as an example. It does not provide low-level monitoring information. At high level, the CloudWatch as its proprietary tool, monitors EC2 and collects VM information. It mainly focuses on timeliness and elasticity, but exhibits the interoperability problem.

As highlighted before, there are various tools and services with different features to monitor clouds. The objective behind their development should address the specific needs of future cloud environments. Such systems must be able to manage and verify a large number of resources and must do it effectively and efficiently. They should provide a solution for different cloud deployment models. Besides, they should be able to effectively spot system impairments and report them to ensure timely interventions, i.e., new resource allocation and service migration. Therefore, monitoring systems must be refined and adapted to different situations of large-scale and highly dynamic environments like clouds.

V. CHALLENGES AND FUTURE WORKS

Although, there is a large body of work considering the development of flexible and self-manageable cloud infrastructures, there still lacks adequate resource management methods, and monitoring systems, e.g., most of the available monitoring systems rely either on grid or service-oriented infrastructures, which are not directly compatible to clouds due to the differences in resource usage models, or due to heavily network-oriented monitoring infrastructures. To achieve more revenue and high resource utilization, service providers have to schedule resources and deploy different user applications but the current scheduling approaches in clouds are biased toward the usage of a single objective, such as execution time instead of considering multiple parameters. Scheduling and deploying multiple applications bring a new set of challenges to the cloud providers. Importantly, for guaranteeing QoS requirements, various changes should be applied in the current scheduling, monitoring, and resource management algorithms. For instance, according to the agreed SLAs, deciding which jobs should be run in the specific context (e.g., peak time) would be another challenge. Furthermore, automatic mechanisms are needed for resource monitoring metrics and preventing SLA violation.

Various types of applications are executed in clouds and each of them has various characteristics and specific requirements. Therefore, frameworks are needed for defining their requirements. The users should assess cloud resources and the feasibility of provisioning their requirements. The providers should be able to meet such requirements. In addition, the expense of using these services as well as the amount of penalties which must be paid if defined QoS metrics cannot be satisfied by the service provider should be negotiated before cloud services are used by users. Finally, the following issues should be addressed properly:

1) SLA Management: Requirements of users can change due to the changes in business operations and operational environments over time. Therefore, service providers should be able to self-configure the reserved resources to satisfy new requirements continuously. Importantly, providers should be

able to automatically decide which requests should be accepted and which one should be rejected. For making these decisions various performance factors such as availability or response time should be considered. Thus, scheduling algorithms and resource provisioning policies should be adjusted such that the requirements of applications with various characteristics are satisfied properly.

2) Defining Requirements: On one hand, tools, protocols, and standards are necessary for describing user requirements. On the other hand, feasibility of provisioning QoS metrics should be evaluated before a service provider accepts a new request.

3) Resource Monitoring: This procedure includes defining and measuring low-level parameters for monitoring performance and availability of resources. By designing an accurate resource monitoring system, providers can have a better understanding of the current state of the system. Thus, they are able to satisfy defined SLA better. Since cloud performance consists of various areas (i.e., elasticity and scalability) with unique functions that can be managed separately, we have identified some of these areas in order to investigate the role of monitoring.

4) Power Consumption Management: Clouds characteristics such as reliability and availability has gained much popularity. The growing demand for these services has increased the power consumption (e.g., computing equipment and the associated cooling infrastructure) in data centers which include thousands of servers. Thus, energy consumption has become a considerable factor in designing modern and green clouds. Consolidating multiple servers running in different VMs on cloud increases the overall utilization and efficiency of the equipment across the whole deployment. Consequently, management, and scheduling of VMs across a cloud in a power-aware fashion is key to reducing the overall operational costs. Current researches on cloud resource management do not sufficiently address the collaboration of minimizing energy cost and maximizing revenue, for various intensive applications. Green computing is defined as environmentally sustainable computing. This term generally refers to the efficient use of computing resources with respect to an environment so that the primary goals such as minimizing environmental impact can be satisfied. There are other goals of green computing, most notably, is to provide computing as a utility with features such as reliability, dynamic and scalable properties with the minimal energy consumption. These goals will not only make the resources more efficient but also enhance the overall performance [100]. One way to reduce the power consumption is to distribute the workload among different servers depending on time and space. Another way to do so is to reduce the CPU utilization and other idle resources. Fewer resources have to be allocated than agreed, but more than actually utilized at the specific point in time, to not violate SLAs [101].

The exponential growth of data volume has led to phenomena known as big data that requires huge computation to be processed. A significant challenge is that this expansion rate of data production surpasses the ability of data processing methods. To meet the processing and storage requirements, cloud is a promising paradigm not only to provide an adequate infrastructure for storing and processing but also for analysis purposes. Given large-scale data, provisioning of virtualized resources, parallel processing, data service integration, and scalable storage are the features which have attracted researchers to optimize processing performance and to minimize storage costs. Hence, big data applications involve reviewed QoS metrics, i.e., scalability and availability. Furthermore, fault tolerance as a provider responsibility, together with elasticity and load balancing, should be provided in a fast pace to fulfill changing application requirements. Efficient management can be guaranteed only by continuously monitoring storage, computation, performance and their respective QoS across different layers.

VI. Conclusion

In this paper, we have reviewed technical details related to cloud QoS, and classified them into specific categories, which offers an opportunity to gain insights into the different aspects of QoS frameworks. From our QoS metrics definition, we have derived a list of capabilities that are considered relevant to facilitate efficient cloud resource management. Additionally, the problem of resource allocation with its different techniques in clouds environment has been considered. Since clouds consist of heterogeneous resources in various areas with unique functions that can be managed separately, we have investigated the pivotal role of QoS monitoring tools from the perspectives of users and providers. We have defined a taxonomy of different capabilities of clouds to provide an in-depth insight into QoS issues in order to achieve a unified framework. Finally we have discussed the challenges and issues being faced by the researchers and practitioners.

References

[1] P. Mell and T. Grance, "The NIST definition of cloud computing," National Institute of Standards and Technology, Information Technology Laboratory, Version 15, Oct. 7, 2009.

[2] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," *Commun. ACM*, vol. 53, no. 4, pp. 50−58, Apr. 2010.

[3] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "Above the clouds: A Berkeley view of cloud computing," EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2009-28, Feb. 10, 2009.

[4] N. Ani Brown Mary and K. Jayapriya, "An extensive survey on QoS in cloud computing," *Int. J. Comp. Sci. Inf. Technol.*, vol. 5, no. 1, pp. 1−5, 2014.

[5] M. Firdhous, S. Hassan, and O. Ghazali, "A comprehensive survey on quality of service implementations in cloud computing," *Int. J. Sci. Eng. Res.*, vol. 4, no. 5, pp. 118−123, 2013.

[6] J. M. Pedersen, M. T. Riaz, J. C. Junior, B. Dubalski, D. Ledzinski, and A. Patel, "Assessing measurements of QoS for global cloud computing services," in *Proc. IEEE 9th Int. Conf. Dependable, Autonomic and Secure Computing*, Sydney, NSW, 2011, pp. 682−689.

[7] R. Buyya, S. K. Garg, and R. N. Calheiros, "SLA-oriented resource provisioning for cloud computing: Challenges, architecture, and solutions," in *Proc. 2011 Int. Conf. Cloud and Service Computing (CSC)*, Hong Kong, China, 2011, pp. 1−10.

[8] A. Stanik, M. Koerner, and O. Kao, "Service-level agreement aggregation for quality of service-aware federated cloud networking," *IET Netw.*, vol. 4, no. 5, pp. 264−269, Sep. 2015.

[9] M. Comuzzi and B. Pernici, "A framework for QoS-based Web service contracting," *ACM Trans. Web (TWEB)*, vol. 3, no. 3, pp. Article ID 10, Jun. 2009.

[10] A. B. Grant and O. T. Eluwole, "Cloud resource management-virtual machines competing for limited resources," in *Proc. 2013 AFRICON, 2013*, Pointe-Aux-Piments, 2013, pp. 1−7.

[11] S. Tang, J. Yuan, C. Wang, and X. Y. Li, "A framework for Amazon EC2 bidding strategy under SLA constraints," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 1, pp. 2−11, Jan. 2014.

[12] D. Armstrong and K. Djemame, "Towards quality of service in the cloud," [Online]. Available: http://www.comp.leeds.ac.uk/ukpew09/papers/18.pdf

[13] Z. Rehman, O. Khadeer Hussain, F. Khadeer Hussain, E. Chang, and T. Dillon, "User-side QoS forecasting and management of cloud services," *World Wide Web*, vol. 18, no. 6, pp. 1677−1716, Nov. 2015.

[14] E. Mostajeran, B. I. Ismail, M. F. Khalid, and H. Ong, "A survey on SLA-based brokering for inter-cloud computing," in *Proc. 2nd Int. Conf. Computing Technology and Information Management (ICCTIM)*, Johor, 2015, pp. 25−31.

[15] A. Maarouf, A. Marzouk, and A. Haqiq, "Towards a trusted third party based on multi-agent systems for automatic control of the quality of service contract in the cloud computing," in *Proc. 2015 Int. Conf. Electrical and Information Technologies (ICEIT)*, Marrakech, 2015, pp. 311−315.

[16] G. Motta, L. L. You, N. Sfondrini, D. Sacco, and T. Y. Ma, "Service level management (SLM) in cloud computing-third party SLM framework," in *Proc. IEEE 23rd Int. WETICE Conf. (WETICE)*, Parma, 2014, pp. 353−358.

[17] A. Keller and H. Ludwig, "The WSLA framework: Specifying and monitoring service level agreements for web services," *Netw. Syst. Manag.*, vol. 11, no. 1, pp. 57−81, Mar. 2003.

[18] A. Andrieux, K. Czajkowski, A. Dan, K. Keahey, H. Ludwig, T. Nakata, J. Pruyne, J. Rofrano, S. Tuecke, and M. Xu, "Web services agreement specification (WS-agreement)," Open Grid Forum (OGF). Accessed on: Mar. 14, 2007. [Online]. Available: https://www.ogf.org/documents/GFD.107.pdf

[19] J. Skene, D. D. Lamanna, and W. Emmerich, "Precise service level agreements," in *Proc. 26th Int. Conf. Software Engineering*, Edinburgh, UK, 2004, pp. 179−188.

[20] H. Goudarzi and M. Pedram, "Multi-dimensional SLA-based resource allocation for multi-tier cloud computing systems," in *Proc. 2011 IEEE Int. Conf. Cloud Computing (CLOUD)*, Washington DC, USA, 2011, pp. 324−331.

[21] T. Chauhan, S. Chaudhary, V. Kumar, and M. Bhise, "Service level agreement parameter matching in cloud computing," *Proc. 2011 World Congr. Information and Communication Technologies (WICT)*, Mumbai, 2011, pp. 564−570.

[22] M. Effatparvar and M. S. Garshasbi, "A genetic algorithm for static load balancing in parallel heterogeneous systems," in *Proc. 2nd Int. Conf. Innovation, Management and Technology Research (ICIMTR)*, Malaysia, 2013, pp. 358−364.

[23] Y. D. Mei, L. Liu, X. Pu, and S. Sivathanu, "Performance measurements and analysis of network I/O applications in virtualized cloud," in *Proc. IEEE 3rd Int. Conf. Cloud Computing*, Miami, FL, USA, 2010, pp. 59−66.

[24] A. Botta, A. Pescapé, and G. Ventre, "Quality of service statistics over heterogeneous networks: Analysis and applications," *Eur. J. Operat. Res.*, vol. 191, no. 3, pp. 1075−1088, Dec. 2008.

[25] S. W. Choi, J. S. Her, and S. D. Kim, "QoS metrics for evaluating services from the perspective of service providers," in *Proc. 2007 IEEE Int. Conf. e-Business Engineering*, Hong Kong, China, 2007, pp. 622−625.

[26] S. K. Garg, S. Versteeg, and R. Buyya, "A framework for ranking of cloud computing services," *Future Gener. Comp. Syst.*, vol. 29, no. 4, pp. 1012−1023, Jun. 2013.

[27] W. Ai, K. L. Li, S. L. Lan, F. Zhang, J. Mei, K. Q. Li, and R. Buyya, "On elasticity measurement in cloud computing," *Sci. Progr.*, vol. 2016, pp. Article ID 7519507, 2016.

[28] J. Y. Lee, J. W. Lee, D. W. Cheun, and S. D. Kim, "A quality model for evaluating software-as-a-service in cloud computing," in *Proc. 7th ACIS Int. Conf. Software Engineering Research, Management and Applications*, Haikou, China, 2009, pp. 261−266.

[29] C. Barba-Jimenez, R. Ramirez-Velarde, A. Tchernykh, R. Rodríguez-Dagnino, J. Nolazco-Flores, and R. Perez-Cazares, "Cloud based video-on-demand service model ensuring quality of service and scalability," *J. Netw. Comp. Appl.*, vol. 70, pp. 102−113, Jul. 2016.

[30] L. Montecchi, N. Nostro, A. Ceccarelli, G. Vella, A. Caruso, and A. Bondavalli, "Model-based evaluation of scalability and security tradeoffs: A case study on a multi-service platform," *Electron. Notes Theor. Comp. Sci.*, vol. 310, pp. 113−133, Jan. 2015.

[31] K. Kritikos, J. Domaschka, and A. Rossini, "SRL: A scalability rule language for multi-cloud environments," in *Proc. IEEE 6th Int. Conf. Cloud Computing Technology and Science (CloudCom)*, Singapore, 2014, pp. 1−9.

[32] M. Beltrán, "BECloud: A new approach to analyse elasticity enablers of cloud services," *Future Gener. Comp. Syst.*, vol. 64, pp. 39−49, Nov. 2016.

[33] D. Jung, S. Chin, K. S. Chung, and H. Yu, "VM migration for fault tolerance in spot instance based cloud computing," in *Grid and Pervasive Computing*, J. J. Park, H. R. Arabnia, C. Kim, W. S. Shi, and J. M. Gil, eds. Berlin Heidelberg: Springer, 2013, pp. 142−151.

[34] M. H. Tu and D. X. Xu, "System resilience modeling and enhancement for the cloud," in *Proc. Int. Conf. Computing, Networking and Communications*, San Diego, CA, 2013, pp. 1021−1025.

[35] B. Sousa, K. Pentikousis, and M. Curado, "Optimizing quality of resilience in the cloud," in *Proc. IEEE Global Communications Conf.*, Austin, TX, 2014, pp. 1133−1138.

[36] J. Nabrzyski, J. M. Schopf, and J. Weglarz, *Grid Resource Management: State of the Art and Future Trends*. US: Springer, 2004.

[37] Q. Cao, Z. B. Wei, and W. M. Gong, "An optimized algorithm for task scheduling based on activity based costing in cloud computing," in *Proc. 2009 Int. Conf. Bioinformatics and Biomedical Engineering*, Beijing, China, 2009, pp. 1−3.

[38] S. Pandey, L. L. Wu, S. Mayura Guru, and R. Buyya, "A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments," in *Proc. 24th IEEE Int. Conf. Advanced Information Networking and Applications*, Perth, WA, 2010, pp. 400−407.

[39] H. T. Yuan, J. Bi, W. Tan, M. C. Zhou, B. H. Li, and J. Q. Li, "TTSA: An effective scheduling approach for delay bounded tasks in hybrid clouds," *IEEE Trans. Cybern.*, vol. PP, no. 99, pp. 1−11, Jul. 2016.

[40] C. Liu, Z. W. Zhao, and F. Liu, "An insight into the architecture of condor-a distributed scheduler," in *Proc. 2009 Int. Symp. Computer Network and Multimedia Technology*, Wuhan, China, 2009, pp. 1−4.

[41] S. Hema and S. Jaganathan, "Improvisioning hadoop in Globus Toolkit," *Proc. 2013 Int. Conf. Circuits, Power and Computing Technologies (ICCPCT)*, Nagercoil, 2013, pp. 1082−1088.

[42] S. N. Zhou, X. H. Zheng, J. W. Wang, and P. Delisle, "Utopia: A load sharing facility for large, heterogeneous distributed computer systems," *Softw. Pract. Exp.*, vol. 23, no. 12, pp. 1305−1336, Dec. 1993.

[43] J. P. Jones, *PBS: Portable Batch System*. Cambridge, MA, USA: MIT Press, 2002, pp. 363−383.

[44] O. Krieger, P. McGachey, and A. Kanevsky, "Enabling a marketplace of clouds: VMware's vCloud director," *ACM SIGOPS Operat. Syst. Rev.*, vol. 44, no. 4, pp. 103−114, Dec. 2010.

[45] J. Appavoo, V. Uhlig, and A. Waterland, "Project kittyhawk: Building a global-scale computer: Blue gene/P as a generic computing platform," *ACM SIGOPS Operat. Syst. Rev.*, vol. 42, no. 1, pp. 77−84, Jan. 2009.

[46] B. Hindman, A. Konwinski, M. Zaharia, A. Ghodsi, A. D. Joseph, R. Katz, S. Shenker, and I. Stoica, "Mesos: A platform for fine-grained resource sharing in the data center," in *Proc. 8th USENIX Conf. Networked Systems Design and Implementation*, Berkeley, CA, 2010, pp. 295−308.

[47] B. Rochwerger, D. Breitgand, E. Levy, A. Galis, K. Nagin, I. M. Llorente, R. Montero, Y. Wolfsthal, E. Elmroth, J. Cáceres, M. Ben-Yehuda, W. Emmerich, and F. Galán, "The reservoir model and architecture for open federated cloud computing," *IBM J. Res. Dev.*, vol. 53, no. 4, pp. 535−545, Jul. 2009.

[48] G. C. Xu, J. J. Pang, X. D. Fu, "A load balancing model based on cloud partitioning for the public cloud," *Tsinghua Sci. Technol.*, vol. 18, no. 1, pp. 34−39, Feb. 2013.

[49] Y. C. Lee, C. Wang, A. Y. Zomaya, and B. B. Zhou, "Profit-driven service request scheduling in clouds," in *Proc. 2010 IEEE/ACM Int. Conf. Cluster, Cloud and Grid Computing*, Melbourne, Australia, 2010, pp. 15−24.

[50] P. Hershey, S. Rao, C. B. Silio, A. Narayan, "System of systems to provide quality of service monitoring, management and response in cloud computing environments," in *Proc. 7th Int. Conf. System of Systems Engineering (SoSE)*, Genoa, 2012, pp. 314−320.

[51] V. C. Emeakaroha, I. Brandic, M. Maurer, and I. Breskovic, "SLA-aware application deployment and resource allocation in clouds," in *Proc. IEEE 35th Ann. Computer Software and Applications Conf. Workshops*, Munich, 2011, pp. 298−303.

[52] B. Farley, A. Juels, V. Varadarajan, T. Ristenpart, K. D. Bowers, and M. M. Swift, "More for your money: Exploiting performance heterogeneity in public clouds," in *Proc. 3rd ACM Symp. Cloud Computing*, New York, NY, USA, 2012, pp. Article ID 20.

[53] J. Schad, J. Dittrich, and J. A. Quiané-Ruiz, "Runtime measurements in the cloud: Observing, analyzing, and reducing variance," *Proc. VLDB Endowment*, vol. 3, no. 1−2, pp. 460−471, Sep. 2010.

[54] A. Khan, X. F. Yan, S. Tao, and N. Anerousis, "Workload characterization and prediction in the cloud: A multiple time series approach," in *Proc. IEEE Network Operations and Management Symp.*, Maui, HI, 2012, pp. 1287−1294.

[55] G. F. Feng, S. Garg, R. Buyya, and W. Z. Li, "Revenue maximization using adaptive resource provisioning in cloud computing environments," in *Proc. ACM/IEEE 13th Int. Conf. Grid Computing*, Beijing, China, 2012, pp. 192−200.

[56] C. Q. Tang, M. Steinder, M. Spreitzer, and G. Pacifici, "A scalable application placement controller for enterprise data centers" in *Proc. 16th Int. Conf. World Wide Web*, New York, NY, USA, 2007, pp. 331−340.

[57] J. Chen, Y. Teofilus Larosa, and P. Yang, "Optimal QoS load balancing mechanism for virtual machines scheduling in eucalyptus cloud computing platform," in *Proc. 2nd Baltic Congress on Future Internet Communications*, USA, pp. 214-221, 2012.

[58] S. Aslanzadeh and Z. Chaczko, "Load balancing optimization in cloud computing: Applying endocrine-particle swarm optimization," in *Proc. 2015 Int. Conf. Electro/Information Technology (EIT)*, Dekalb, IL, 2015, pp. 165−169.

[59] K. Dasgupta, B. Mandal, P. Dutta, J. K. Mondal, and S. Dam, "A genetic algorithm (GA) based load balancing strategy for cloud computing," in *Proc. 1st Int. Conf. Computational Intelligence: Modeling Techniques and Applications (CIMTA)*, Kalyani, India, 2013, pp. 340−347.

[60] D. Kashyap and J. Viradiya, "A survey of various load balancing algorithms in cloud computing," *Int. J. Sci. Technol. Res.*, vol.3, no. 11, pp. 115−119, Nov. 2014.

[61] A. Sadeghi Milani and N. Jafari Navimipour, "Load balancing mechanisms and techniques in the cloud environments: Systematic literature review and future trends," *J. Netw. Comp. Appl.*, vol. 71, pp. 86−98, Aug. 2016.

[62] A. Rahman, X. Liu, and F. X. Kong, "A survey on geographic load balancing based data center power management in the smart grid environment," *IEEE Commun. Surv. Tutor.*, vol. 16, no. 1, pp. 214−233, 2014.

[63] M. Kalra and S. Singh, "A review of metaheuristic scheduling techniques in cloud computing," *Egypt. Inf. J.*, vol. 16, no. 3, pp. 275−295, Nov. 2015.

[64] S. G. Domanal and G. R. Mohana Reddy, "Optimal load balancing in cloud computing by efficient utilization of virtual machines," in *Proc. 6th Int. Conf. Communication Systems and Networks (COMSNETS)*, Bangalore, 2014, pp. 1−4.

[65] M. Katyal and A. Mishra, "Application of selective algorithm for effective resource provisioning in cloud computing environment," *Int. J. Cloud Comput. Serv. Archit.*, vol. 4, no. 1, Feb. 2014, doi: doi:10.5121/ijccsa.2014.4101.

[66] K. Al Nuaimi, N. Mohamed, M. Al Nuaimi, and J. Al-Jaroodi, "A survey of load balancing in cloud computing: challenges and algorithms," in *Proc. IEEE 2nd Symp. Network Cloud Computing and Applications*, London, 2012, pp. 137−142.

[67] G. Patel, R. Mehta, and U. Bhoi, "Enhanced load balanced min-min algorithm for static meta task scheduling in cloud computing," in *Proc. 3rd Int. Conf. Recent Trends in Computing*, Modinagar, India, 2015, pp. 545−553.

[68] B. G. Batista, J. C. Estrella, M. J. Santana, R. H. C. Santana, and S. Reiff-Marganiec, "Performance evaluation in a cloud with the provisioning of different resources configurations," in *Proc. 2014 IEEE World Congr. Services*, Anchorage, AK, 2014, pp. 309−316.

[69] B. S. Rajeshwari and M. Dakshayini, "Optimized service level agreement based workload balancing strategy for cloud environment," in *Proc. 2015 IEEE Int. Advance Computing Conf. (IACC)*, Banglore, 2015, pp. 160−165.

[70] E. Imamagic and D. Dobrenic, "Grid infrastructure monitoring system based on Nagios," in *Proc. 2007 Workshop on Grid Monitoring*, New York, NY, USA, 2007, pp. 23−28.

[71] W. Chen, X. Q. Qiao, J. Wei, and T. Huang, "A profit-aware virtual machine deployment optimization framework for cloud platform providers," in *Proc. IEEE 5th Int. Conf. Cloud Computing*, Honolulu, HI, 2012, pp. 17−24.

[72] J. Z. Liu, Y. X. Zhang, Y. Z. Zhou, D. Zhang, and H. Liu, "Aggressive resource provisioning for ensuring QoS in virtualized environments," *IEEE Trans. Cloud Comput.*, vol. 3, no. 2, pp. 119−131, Apr.-Jun. 2015.

[73] L. L. Wu, S. K. Garg, and R. Buyya, "SLA-based resource allocation for software as a service provider (SaaS) in cloud computing environments," *Proc. 11th IEEE/ACM Int. Symp. Cluster, Cloud and Grid Computing*, Newport Beach, CA, 2011, pp. 195−204.

[74] I. Fujiwara, K. Aida, and I. Ono, "Market based resource allocation for distributed computing," Information Processing Society, Japan, IPSJ SIG Tech. Rep. 1, 2009.

[75] M. Unuvar, S. Tosi, Y. N. Doganata, M. Steinder, and A. N. Tantawi, "Selecting optimum cloud availability zones by learning user satisfaction levels," *IEEE Trans. Serv. Comput.*, vol. 8, no. 2, pp. 199−211, Mar.-Apr. 2015.

[76] S. Ding, C. Y. Xia, Q. Cai, K. L. Zhou, and S. L. Yang, "QoS-aware resource matching and recommendation for cloud computing systems," *Appl. Math. Comput.*, vol. 247, pp. 941−950, Nov. 2014.

[77] J. Bi, H. T. Yuan, W. Tan, M. C. Zhou, Y. S. Fan, J. Zhang, and J. Q. Li, "Application-aware dynamic fine-grained resource provisioning in a virtualized cloud data center," *IEEE Trans. Autom. Sci. Eng.*, 2015, doi: 10.1109/TASE.2015.2503325, to be published.

[78] Q. Duan, Y. Yan, and A. V. Vasilakos, "A survey on service-oriented network virtualization toward convergence of networking and cloud computing," *IEEE Trans. Netw. Serv. Manag.*, vol. 9, no. 4, pp. 373−392, Dec. 2012.

[79] J. Huang, Q. Duan, S. Guo, Y. H. Yan, and S. Yu, "Converged network-cloud service composition with end-to-end performance guarantee," *IEEE Trans. Cloud Comput.*, 2015, doi: 10.1109/TCC.2015.2491939, to be published.

[80] W. Fu and Q. Huang, "Grideye: A service-oriented grid monitoring system with improved forecasting algorithm," in *Proc. 5th Int. Conf. Grid and Cooperative Computing Workshops*, Hunan, China, 2006, pp. 5−12.

[81] D. Gunter, B. Tierney, B. Crowley, M. Holding, and J. Lee, "Netlogger: A toolkit for distributed system performance analysis," in *Proc. 8th Int. Symp. Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, San Francisco, CA, 2000, pp. 267−273.

[82] T. Wood, P. Shenoy, A. Venkataramani, and M. Yousif, "Sandpiper: Black-box and gray-box resource management for virtual machines," *J. Comp. Netw.*, vol. 53, no. 17, pp. 2923−2938, Dec. 2009.

[83] P. C. Hershey, S. Rao, C. B. Silio, and A. Narayan, "System of systems for quality-of-service observation and response in cloud computing environments," *IEEE Syst. J.*, vol. 9, no. 1, pp. 212−222, Mar. 2015.

[84] K. Ma, R. Sun, and A. Abraham, "Toward a lightweight framework for monitoring public clouds, in *Proc. Fourth International Conference on Computational Aspects of Social Networks (CASoN)*, Carlos, Brazil, pp. 361-365, 2012.

[85] H. L. Yeh, Y. F. Chen, T. T. Yeh, P. C. Huang, S. H. Liu, H. W. Wei, and T. S. Hsu, "A monitoring system based on Nagios for data grid environments," in *Proc. 2011 World Congr. Computer Science, Computer Engineering, and Applied Computing*, San Diego, USA, 2011.

[86] M. L. Massie, B. N. Chun, and D. E. Culler, "The ganglia distributed monitoring system: Design, implementation, and experience," *Parallel Comput.*, vol. 30, no. 7, pp. 817−840, Jul. 2004.

[87] A. Iosup, S. Ostermann, M. N. Yigitbasi, R. Prodan, T. Fahringer, and D. Epema, "Performance analysis of cloud computing services for many-tasks scientific computing," *IEEE Trans. on Parallel and Distrib. Syst.*, vol. 22, no. 6, pp. 931−945, Jun. 2011.

[88] S. Clayman, A. Galis, C. Chapman, G. Toffetti, L. Rodero-Merino, L. M. Vaquero, K. Nagin, and B. Rochwerger, "Monitoring service clouds

in the future internet," in *Towards the Future Internet-Emerging Trends from European Research*, G. Tselentis, A. Galis, A. Gavras, S. Krco, V. Lotz, E. Simperl, B. Stiller, and T. Zahariadis, eds. IOS Press, 2010, pp. 115−126.

[89] Y. Huang, H. Su, W. Sun, J. M. Zhang, C. J. Guo, J. M. Xu, Z. B. Jiang, S. X. Yang, and J. Zhu, "Framework for building a low-cost, scalable, and secured platform for web-delivered business services," *IBM J. Res. Dev.*, vol. 54, no. 6, pp. 4:1−4:14, Nov.-Dec. 2010.

[90] C. Bunch, V. Arora, N. Chohan, C. Krintz, S. Hegde, and A. Srivastava, "A pluggable autoscaling service for open cloud PaaS systems," in *Proc. IEEE 5th Int. Conf. Utility and Cloud Computing (UCC)*, Chicago, IL, 2012, pp. 191−194.

[91] I. Bermudez, S. Traverso, M. Mellia, and M. Munafó, "Exploring the cloud from passive measurements: The Amazon AWS case," in *Proc. IEEE INFOCOM*, Turin, 2013, pp. 230−234.

[92] E. Roloff, F. Birck, M. Diener, A. Carissimi, and P. O. A. Navaux, "Evaluating high performance computing on the Windows Azure platform," *Proc. 5th Int. Conf. Cloud Computing (CLOUD)*, Honolulu, HI, 2012, pp. 803−810.

[93] B. Ferriman, T. Hamed, and Q. H. Mahmoud, "Storming the cloud: A look at denial of service in the Google App Engine," in *Proc. 2015 Int. Conf. Computing, Networking and Communications (ICNC)*, Garden Grove, CA, 2015, pp. 363−368.

[94] L. X. Fu and C. Gondi, "Cloud computing hosting," in *Proc. 3rd IEEE Int. Conf. Computer Science and Information Technology (ICCSIT)*, Chengdu, China, 2010, pp. 194−198.

[95] C. Heyer, D. Homoki, K. Lakshminarayanan, and T. Whiffen, "Maximizing the use of VMware vRealize operations for horizon," Tech. White Paper. [Online]. Available: https://www.dropbox.com/s/xectcatxnrrerfr/vmware-white-paper.pdf?dl=0

[96] K. Fatema, V. C. Emeakaroha, P. D. Healy, J. P. Morrison, and T. Lynn, "A survey of cloud monitoring tools: Taxonomy, capabilities and objectives," *J. Parallel Distrib. Comput.*, vol. 74, no. 10, pp. 2918−2933, Oct. 2014.

[97] S. Ostermann, A. Iosup, N. Yigitbasi, R. Prodan, T. Fahringer, and D. Epema, "A performance analysis of EC2 cloud computing services for scientific computing," in *Cloud Computing*, D. R. Avresky, M. Diaz, A. Bode, B. Ciciani, and E. Dekel, eds. Berlin Heidelberg: Springer, 2010, pp. 115−131.

[98] Y. Xia, M. C. Zhou, X. Luo, S. Pang, Q. Zhu, and J. Li, "Stochastic modeling and performance analysis of migration-enabled and error-prone clouds," *IEEE Trans. on Industrial Informatics*, vol. 11, no. 2, pp. 495-504, Apr. 2015.

[99] A. Datt, A. Goel, and S. C. Gupta, "Monitoring list for compute infrastructure in Eucalyptus cloud," in *Proc. 24th Int. Conf. Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*, Larnaca, 2015, pp. 69−71.

[100] M. Ghamkhari and H. Mohsenian-Rad, "Energy and performance management of green data centers: A profit maximization approach," *IEEE Trans. Smart Grid*, vol. 4, no. 2, pp. 1017−1025, Jun. 2013.

[101] C. Valliyammai, S. Uma, K. Dhivya Bharathi, and P. Surya, "Efficient energy consumption in green cloud," in *Proc. Int. Conf. Recent Trends in Information Technology (ICRTIT)*, Chennai, 2014, pp. 1−4.

Mohammad Hossein Ghahramani obtained the B.S. degree and M.S. degree in information technology engineering from Amirkabir University of Technology-Tehran Polytechnic, Iran. He was the technical manager of Information Center of Institute for Research in Fundamental Sciences from 2008 to 2014. He is currently working toward the Ph.D. degree at Macau University of Science and Technology. His research interests are focused on big data analysis and application of cloud computing.

MengChu Zhou (S88-M90-SM93-F03) received his B.S. degree in control engineering from Nanjing University of Science and Technology, Nanjing, China in 1983, M.S. degree in automatic control from Beijing Institute of Technology, Beijing, China in 1986, and Ph.D. degree in computer and systems engineering from Rensselaer Polytechnic Institute, Troy, NY in 1990. He joined New Jersey Institute of Technology (NJIT), Newark, NJ in 1990, and is now a distinguished professor of electrical and computer engineering. His research interests are in Petri nets, internet of things, big data, web services, manufacturing, transportation, and energy systems. He has over 680 publications including 12 books, 360+ journal papers (260+ in IEEE Transactions), and 28 book-chapters. He is the founding editor of IEEE Press Book Series on Systems Science and Engineering. He is a recipient of Humboldt Research Award for US Senior Scientists, Franklin V. Taylor Memorial Award and the Norbert Wiener Award from IEEE Systems, Man and Cybernetics Society. He is a life member of Chinese Association for Science and Technology-USA and served as its president in 1999. He is a fellow of International Federation of Automatic Control (IFAC) and American Association for the Advancement of Science (AAAS). Corresponding author of this paper.

Chi Tin Hon received his Ph.D. degree in management science and engineering from South China University of Technology, Guangzhou, China in 2008, did his postdoc study in management science and engineering from Fudan University, Shanghai, China in 2013, and has been visiting researcher of Laboratory for Discrete Event Systems, New Jersey Institute of Technology (NJIT), Newark, NJ in 2015. His research interests are in system engineering, information security, eBusiness and big data. His patent and technology application has been awarded for APICTA Merit Award in 2014 and 2015 and he received Macau Science and Technology Invention Award in 2012 and 2016. He is a Senior Member of IEEE.