# Service Quality Assurance in Multi-Clouds *

Dana Petcu

**Abstract**

A particular problem of cloud environments is the assurance of a certain level of service quality. The problem is escalated in the case of building support platforms for using multiple clouds. Various partial solutions to ensure a certain quality level of the cloud services have been investigated in the last half decade. This paper analyzes the existing approaches to define, model, evaluate, estimate, measure of optimize the quality of services offered to cloud-based applications. A particular approach is detailed, the one that uses model-driven engineering techniques. Moreover, the special case of designing data-intensive applications, the appropriate quality of service attributes are identified.

## 1 Introduction

Service quality assurances are expected to reduce risks as well as allow the increase of transaction volume and thus expanding the services' market as a whole. Service quality assurances are essential in the nowadays' on-line world. Their mechanisms are causing the services with the highest quality to prosper and those having low service quality to disappear [24]. Therefore, service quality assurance can be seen as a catalyst of the evolution toward efficient service markets and customer high satisfaction.

This paper presents an overview of the current status of the quality of services (QoS) characteristics, assessment and evaluation in cloud environments. A particular attention is provided to the case of multi-clouds and the latest mechanisms for QoS assurance inspired from model-driven engineering techniques.

The paper is organized as follows. The next section is looking to the QoS definition, relationship with SLAs, evaluation and models. The third section is presenting the multi-clouds needs in terms of QoS, proposes a taxonomy, discusses the QoS procedures for service selection and composition, and underlines the need of continuous QoS assurance and optimization. The fourth section is focusing on the model-driven engineering approach for QoS assurance in multi-clouds. The fifth section is discussing the particular case of QoS in data-intensive application engineering.

## 2 QoS in Clouds

**Definition.** The Quality of Service (QoS) was defined in [16] in 2008 as the totality of characteristics of a service that bear on its ability to satisfy stated and implied needs of the user of the service. QoS is an inherent element of service-oriented architecture. Nowadays, QoS play a major role in cloud environments in order to implement the concept of measured service [18].

The QoS is determined by the fulfillment of both functional and non-functional requirements [30]. Meeting the user's requirements with regard to the functionality depends on the service's description. The amount of non-functional quality attributes that have to be considered is very high in the case of cloud services. Therefore, often QoS parameters are considered to be related to non-functional properties of a cloud service.

**Relationship with performance.** The performance is the key QoS attribute that is important to both users and providers of cloud services [9]. Cloud end users are maily interested in the time behaviour of the application running on top of cloud services (like the response time, processing time, or throughput). Another concern is the resource

*This is a pre-print of the paper to appear in GECON 2015 volume at Springer Verlag (invited talk)

consumption and techniques like monitoring resource utilization to identify over-provisioned, under-performance. Reliability or consumption pattern identification are essential to maintain a certain level of QoS. Guaranteed performance in the cloud is allowing to protect the rights of clients and of their suppliers [22].

**Relationship with SLA.** The QoS attributes are usually specified in Service Level Agreements (SLA). The SLA is an adequate solution to specify the QoS guarantees, as it specifies the service level objectives to ensure that the delivered QoS meets the user expectations.

**Relationship with monitoring.** The QoS attributes like response time or throughput have a strong variability and in order to implement the contract, these parameters need to be carefully controlled. Consequently, continuous supervision of QoS attributes is necessary to honor the SLAs by the service provider [7]. SLA compliance is also often tested by the clients. Monitoring of the various parameters of SLAs is a common practice to ensure the compliance with negotiated terms. The monitoring the QoS agreements allows to observe the behaviour of the service and it is based on extracting metrics needed to make measurements of QoS. In [10, 20], for example, the servers, the application, the databases and networks are monitored using agent technologies. The processes associated with monitoring that ensure prevention, correction, and control remain key issues to properly solve the trade-off between the benefit of the supplier and the customer satisfaction [22].

**Optimality: a trade-off between user requirements and provider offers.** An important challenge for cloud providers is to automate the management of cloud resources while keeping into account the QoS requirements of hosted applications as well as the resource supervision expenses [7]. For example, the paper [21] proposes a QoS-aware placement of virtual machines by using a polynomial time heuristic method that solves the problem defined as an integer linear programming model. Another paper, [8], proposes a negotiation mechanism for assuring the QoS requirements that can identify the most optimal agreement in a search space with a large number of parameters. It is based on a co-evolutionary multi-agent approach based on preference ordering.

The authors of [7] identified the quality attributes based on customers desires associated with SLA, as well as the metrics to measure the deviation of QoS from predictables, with possible resolution in the outline of architecture for spontaneous supervision of QoS without violation of SLA.

**QoS evaluation.** The main challenge for the QoS evaluation is related to the fact that the performance in cloud environments (especially the response time) is varying in time due to the resource contention of concurrent competing requests. The run-time performance related parameters such as availability are changing in time. Therefore it is difficult to assure the accurate evaluation of QoS [34]. QoS should be maintained not only in the presence of background dynamic resource provisioning but also in the presence of variable capacity of the servers.

The use of quantitative indexes is an accepted practice. Usually a set of indicators is compiled in just one value [18]. In this context, the Service Measurement Index (SMI) is a clear candidate to assess the QoS. SMI [11] is designed to enable the analysis and assessment of cloud services before using a service as well as while using the service. The domains of QoS covered by the SMI are accountability, agility, assurance, financial, performance, security, privacy, usability [36].

**Models.** There are several reported efforts trying to model QoS in clouds or to manage non-functional properties in an intelligent fashion. The main objective of these QoS models is to support users evaluating the quality of their cloud services [30]. However, the lack of standards for QoS attributes in cloud environments is hindering the deployment of advanced techniques for QoS management [18].

The paper [1] provides an overview of research works in the cloud QoS modeling space. The survey considers QoS modeling techniques for interactive cloud services (e.g. multi-tier applications) and focus on QoS aspects related to performance, reliability and availability.

A specific direction of research consists in the definition of QoS attributes in cloud environments. The paper [15] describes the mathematical model for the QoS metrics, focusing on the QoS categories of performance and security. For performance, these metrics are delay, delay variation, throughput, information overhead. For the security, the metrics

are authentication, authorization, accreditation, integrity, information availability, certification, physical security, non-repudiation. The authors of [30] extracted over two hundred relevant attributes from seven existing cloud QoS models and fifty quality aspects. The most frequently mentioned quality criteria have lead to so-called quality dimensions: reliability, flexibility, performance, data security, costs and conditions, usability, customer service, and provider's sense of responsibility.

Another concern is related to the QoS model life-cycle. The paper [13] focus on a Quality Model life-cycle that has several stages: strategy, design, transition, operation, continual improvement. Two quality models are defined: one for clients and other for providers. The authors of [36] synthesizes the existing research on approaches to the quality analysis of cloud services and they reveals multiple quality analysis approaches that focus either on specific quality aspects or on a specific step in the activity cycle. One accepted fact is that the QoS consists of three dimensions: the technical quality of the outcome, the functional quality of the process and the image of the service provider.

**Final remarks.** Despite the multiple efforts which partially mentioned above, QoS assurance remains an open problem in cloud environment both from theoretical and practical point of view. With the advent of using services from multiple clouds, the complexity of the problem increases.

# 3 QoS in Multi-Clouds

**Multi-cloud context.** The multiple clouds usage scenarios are referring to the serial actions, when applications are moved from one cloud to another, or simultaneous actions, when an application is using services from various clouds. The simplest scenarios are the migration from a private cloud to a public cloud, respectively when some services are lying on the private cloud, while other services are lying on a public cloud (hybrid cloud). The reasons for using multiple clouds are various, from cost changes, availability problems, avoidance of lock-in, national law constraints, peaks, offer enhancement, service particularities, and not at least the perceived QoS [25].

The multi-cloud is a delivery model for multiple clouds which assumes that there is no priori agreement between the cloud providers and a third party is responsible for the services contacts the service providers, negotiates the terms of service consumption, monitors the fulfillment of the service level agreements, triggers the migration of codes, data and networking from one provider to another [25].

In multi-clouds the main problem is the portability [26]. Other issues are related to automated deployment, service aggregation, governance, service selection mechanism and methodology, search engines and so on. The main requirements were exposed for example in [25].

**Taxonomy.** We propose in Figure 1 a taxonomy of the QoS attributes in multi-clouds. The QoS attributes included under flexibility category are of particular importance in the case of multi-cloud, by defining their reason to be.

**QoS management platforms.** QoS management is usually interpreted as the process of allocating resources to the application to guarantee a SLA parameters as performance, availability or reliability. It needs an intelligent environment of self-management application components based on domain knowledge in which application components can be optimized thus facilitating the transition to an advanced governance environment. Reactive control systems taking into QoS attributes are needed to provide an intelligent cloud resource management. A QoS API is expected to be defined, maybe using the SMI indicators, in order to be able to implement a real cloud QoS management platform [18].

Cloud management platforms are currently commonly used in managing the different layers of cloud-based applications. In the ecosystem of RightScale, Enstratus, Cloudability, Cloudyn or CloudExpress, the clients can take advantage of added-value services using a common API that represent a major effort to unify information exposed by providers [18]. However, the multi-clouds supporting platforms do not had until now a clear objective to support QoS management.

The paper [32] proposed a framework named DynaQoS to provide QoS-guaranteed automatic resource management including support for multiple-objective control and service differentiation, as well as a self-tuning fuzzy controller with flexible rule selection.

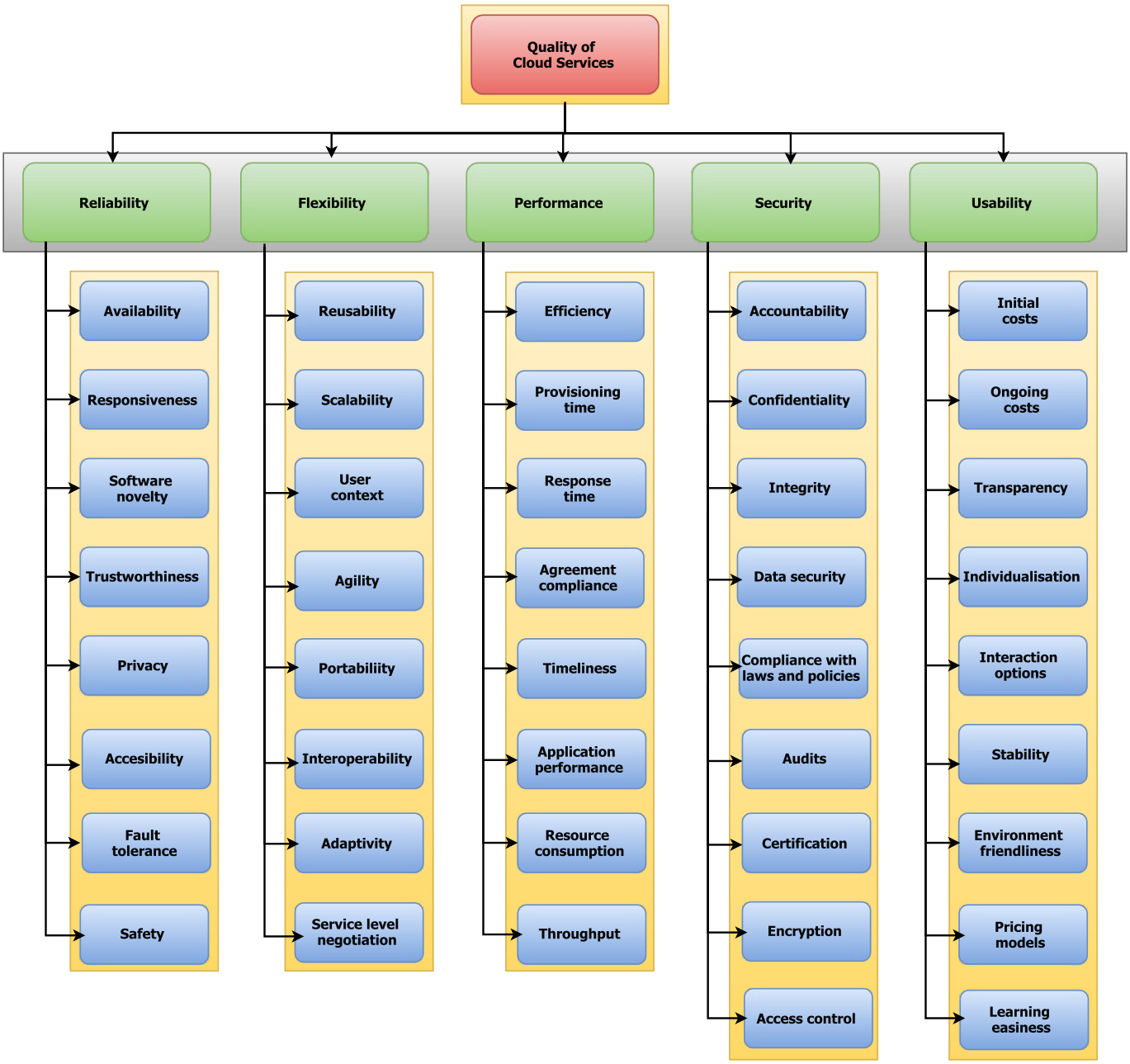


Figure 1: QoS attributes (non-functional) in multi-clouds

**Service selection.** Equal QoS is not possible to be attained by all cloud service providers [34]. The QoS of some providers are superior in performance, others in security or in offering the minimum costs. Consequently, the cloud users should decide the services that are appropriate for their applications, or they should appeal to human or semi-automated expert systems to do that.

The paper [27] discuss a set of QoS-aware brokering strategies, in the particular case of hybrid clouds, which aim to maximize the user satisfaction, cloud broker revenues and reduce energy costs.

The authors of [23] address the problem of QoS ranking prediction for the order of services under considerations for a particular user by proposing a particle swarm optimization algorithm that measure the similarity between two users by considering the occurrence probability of service pairs.

In the experiments reported in [34] the performance evaluation of cloud service providers was combined with monitored QoS data in order to obtain an accurate evaluation of QoS for cloud users. In particular, a fuzzy synthetic decision is used to evaluate the cloud service providers according to the users' preferences, and, then, based on monitored QoS data, a specific model is used to calculate the uncertainty of the cloud services. Then, fuzzy logic control is used to obtain QoS evaluation.

Smart CloudBench [9] is a system that offers automated, on-demand, real-time and customized benchmarking of software systems deployed on cloud IaaS. It is based on an analysis of run-time behavior of cloud services. It helps its users to compare available offerings in the cloud selection phase and to monitor performance for QoS assurance in the cloud consumption phase. The performance tests are referring to load, scalability, endurance, stress and spike.

The QoS attributes like trust, reliability or security can be modeled and discovered in automatic ways [18]. However, in these cases it is also required to take into account the user feedback. Therefore, the complete automation of a broker service is not possible as human validation is required to make strategic decisions.

**Service composition.** QoS is increasingly significant when composing services because a degrading QoS in one of the services can dangerously disturb the QoS of the complete composition [7]. To tackle with this issue, the authors of [17] proposes building a data warehouse of QoS to achieve better service matching and enhance dynamic service composition. A centralized storage combines QoS information from various cloud sources and helps solve the cloud service selection problem through processing of large numbers of historical QoS of cloud services. The proposed QoS data warehouse model supports a deep analysis of the services interior structure and properties through online database analysis, reasoning about complex service weakness points, as well as visual representation of analysis results. The QoS attributes that are analyzed are availability, response time, documentation, best practice, throughput, latency, successability, reliability, and compliance.

**Continuous QoS assurance and optimization.** The cloud services evolve over time, in terms of provision, APIs, performance or availability. In these conditions, it is difficult to assess if a service keeps the compliance with current policies and regulations, with the technical specifications or SLAs, or fulfills functional and non-functional requirements of a user application. The exponential increase of the number of cloud services with similar functionality contribute also to the problem complexity, forcing the clients to invest considerable efforts in identifying the optimum cloud service for their application [19]. As continuous quality assurance and optimization of cloud is impossible for individual consumers, this reality creates the opportunities for a market of cloud service intermediaries, like the third party supporting a multi-cloud. Moreover, continuous QoS assurance and optimization is a highly required feature of cloud broker component of the multi-clouds.

Intermediation for continuous QoS assurance represents an open research topic. The paper [4] focuses on two specific forms of continuous QoS assurance intermediation, policy-driven cloud service governance and service level failure mitigation for cloud services. The scenario provided as example demonstrates the utility of continuous quality assurance intermediation capability.

The technology enabling to support continuous QoS assurance and optimization brokerage is already available [19]. The relevant tools that can provide the building blocks for an implementation of such continuous QoS assurance and optimization in brokers are the tools for monitoring and controlling services, applications, virtualised infrastructures, or even tools for integrating applications, processes and data.

In this context, the paper [19] presents the essential requirements for a software framework enabling continuous QoS assurance and optimization. Broker@Cloud is a software framework fulfilling these requirements [32]: the framework allows governance and quality control, failure prevention and recovery, as well as optimization. The broker capabilities are related to service discovery, integration, aggregation, customization, quality assurance, and optimization. Moreover, the work presented in [31] establishes a platform-agnostic ontology for the facilitation of QoS assurance brokerage.

# 4 QoS Assurance via Model-driven Engineering

**Model-driven techniques for multi-clouds.** Model-driven development techniques allow the shift of the programming paradigms from code-centric to model-centric. Models are the objects of the development process enabling the developers to work at a level of abstraction that allows them to focus on cloud adoption rather on implementation specifics. Moreover, the model transformations help the automation of the process of going from abstract concepts to implementation. This approach is highly relevant for the design and management of applications across multiple clouds. Furthermore, models can also be used to reason about the application QoS and to support design-time exploration methods that identify the best cloud deployment configuration that satisfy certain QoS constraints.

In this context, the main goal of MODAClouds project[1] was to provide methods, a decision support system, an open source integrated development environment (IDE), a run-time environment for the high-level design, early prototyping, semi-automatic code generation, and automatic deployment of applications on multi-clouds with guaranteed QoS. MODAClouds envisioned to design abstractions that help a QoS engineer to specify non-functional requirements and tools to evaluate and compare multiple cloud architectures according their availability, cost and performance.

**MODAClouds' QoS models.** At the highest level of abstraction, independent from the cloud, the QoS model includes information concerning expected QoS characteristics at the application level. QoS contraints can be attached to specific application components or services. At the lowest level, closer to the cloud environment, the QoS model includes data concerning QoS characteristics of the specific cloud resources.

The overall process for applying the MODAClouds' model-based approach is the following. The starting point for QoS analysis is at a first level of description of the target application, a level that is independent from the cloud. The QoS engineer specifies QoS constraints at the this level (e.g., constraints on the application response times). QoS constraints are then semi-automatically translated into monitoring rules and additional monitoring rules can be also specified. At the next level, dependent on the cloud, but independent from a specific provider, constraints or monitoring rules predicating on specific cloud computing concepts can be included (e.g., constraints specifying that the CPU utilization of an application container must be lower than a given threshold). The same happens for the what concerns the cloud provider dependent level: QoS constraints (like monitoring rules) can be inherited from the previous level and new constraints (rules) predicating on cloud specific model concepts can be introduced (e.g. cost information that offers the possibility to estimate the application costs). Monitoring rules at all levels are feed into a monitoring engine. Then the QoS Engineer can define the design alternatives and optimization constraints to perform the performance assessment of a given candidate solution. Finally, when the application is fully deployed, design time parameters can be refined through the analysis of run-time monitoring data.

In the early stage of the project, the authors of [2] described the state of the art regarding cloud QoS modeling concepts and performance annotations and evaluation of design-time models. The document specifies the requirements for QoS evaluation, design-time exploration and optimization tools developed in the frame of MODAClouds project. The investigated attributes of QoS are performance, cost and availability. The key metrics for performance were throughput, response time and system utilization. A special attention was given to model-based performance prediction.

The stochastic models considered suitable for performance analysis are Markovian chains, queueing systems, layered queueing networks, mean field model, fluid queues, or Petri nets. Such models are coupled with design exploration methods based on local search in order to examine design alternatives through what-if analyses. A QoS engineer is expected to express performance queries formulated upon QoS models (such queries allowing to verify the correctness of the performance requirements in the design phase). Quality-driven model transformation approaches can be used to navigate the design space or to specify feedback rules.

The QoS models developed in MODAClouds are based on layered queueing network models, which capture the contention between users for the available hardware and software resources, and the interaction between them. To parameterize these models, it is essential to estimate the inter-request times and the resource consumption associated with each request. Inter-request times can be extracted from the information and the data that is typically tracked by application-logs or container-level logs. Resource consumptions are harder to obtain as not being tracked by logs, and the deep monitoring instrumentations typically required pose unacceptably large overheads.

[1] www.modaclouds.eu

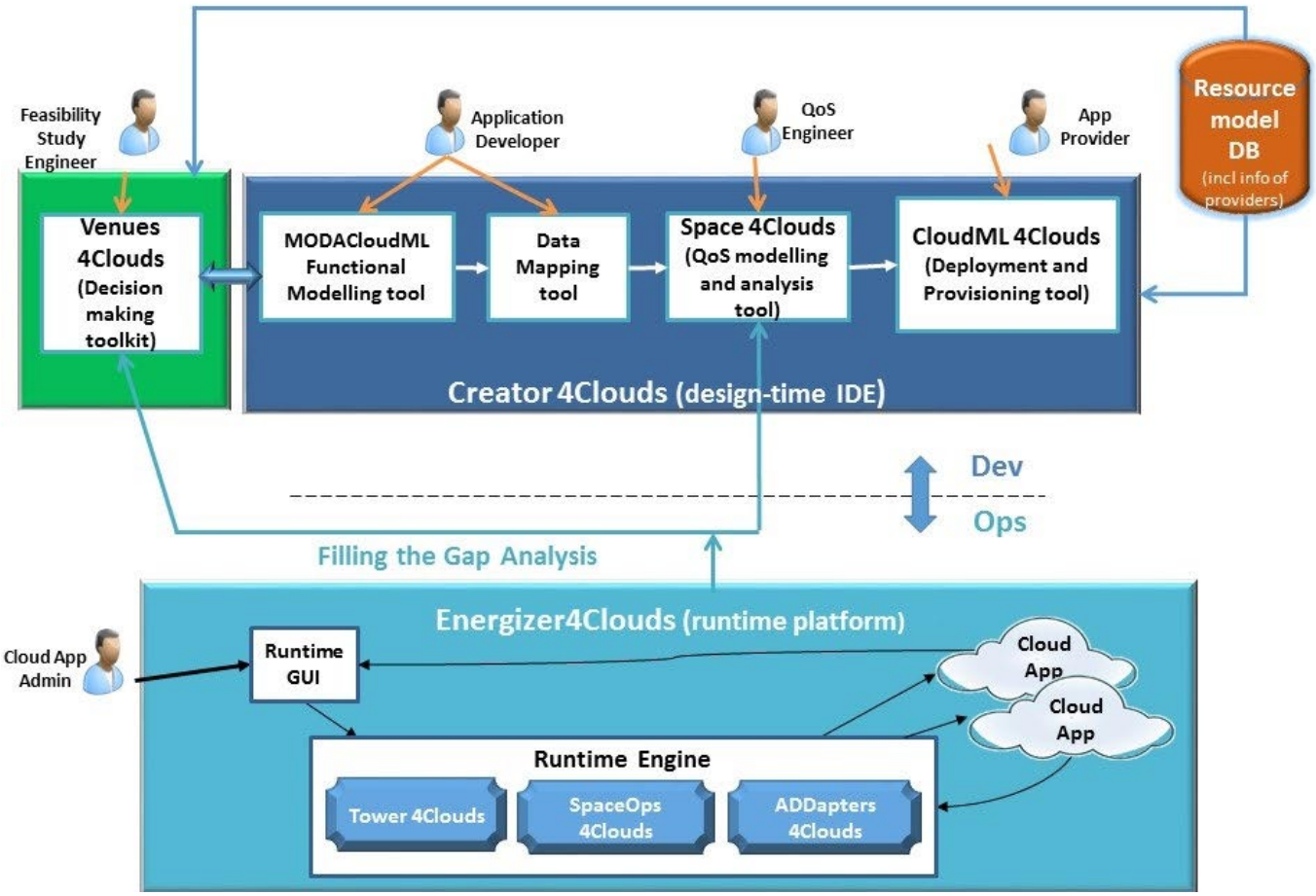


Figure 2: Workflow of the MODAClouds Toolbox

**MODAClouds Toolbox.** The MODAClouds model-driven approach is supported by the MODAClouds Toolbox (Figure 2). This has three main components[2]: an IDE for high-level application design (Creator 4Clouds), a decision support system that helps decision makers identify and select the best execution venue for cloud applications, by considering technical and business requirements (Venues 4Clouds), and a multi-cloud run-time environment energized to provide automatic deployment and execution of applications with guaranteed Quality of Service (QoS) on compatible multi-clouds (Energizer 4Clouds).

Creator 4Clouds provides features to assess the QoS guarantees required by the application. It allows to analyze the QoS trade-offs of various possible application configurations (Space$^{Dev}$4Clouds), map high level data models into scalable NoSQL, and deploy the resulting application on multi-clouds by exploiting the CloudML[3] language. In particular, MODACloudML, a specific extension of CloudML, allows the definition of QoS constraints. A constraint is classified in hard or soft: hard constraints can never be violated, while soft constraints have certain priorities (the constraints with the highest associated priority are the ones more likely to be satisfied). The priority value associated to soft constraints are used for design time exploration [3].

In Space$^{Dev}$4Cloud a two-step approach for search space exploration has been developed. In the first step, an initial configuration of the services is derived automatically using a partially specified application description provided by a QoS engineer. The solution is based on approximated performance models: the QoS associated to a deployment solution is calculated by means of a queuing model with processor sharing policy. In the second step, a local-search-

[2]Available at http://www.modaclouds.eu/software/open-source-repositories/ as open-source codes
[3]www.cloudml.org

based optimization algorithm iteratively improves the starting cloud deployment exploring several configurations. A performance model layered queueing network is employed to derive more accurate estimates of the QoS.

Energizer 4Clouds includes the frameworks to support monitoring (through Tower 4Clouds) and self-adaptation (Space$^{Ops}$4Clouds), together with utilities that perform auxiliary tasks in the platform (ADDapters 4Clouds). Through Tower 4Clouds, operators are able to perform complex monitoring and data analyses from multiple sources.

Space$^{Ops}$4Clouds identifies and actuates proper self-adaptation actions that take into account the current and foreseen state of the system under control. QoS constraints at run-time are translated into monitoring rules (inputs for the monitoring platform that is part of the Tower 4Clouds).

Monitoring rules are specifying the aggregation process of the fine grained monitoring data coming data collectors, under what conditions an alarm is raised (a condition can be a constraint on the values of the monitored data), and what actions are taken in case an alarm is triggered.

MODAClouds IDE can automatically generate monitoring rules from QoS constraints, while the QoS engineer can modify or add new rules. The QoS engineer specifies the constraints, delegates the creation of monitoring rules to the IDE, and then can modify or creates new rules. The framework allows to manipulate rules, as well as constraints, through a visual representation.

Space 4Clouds ($^{Dev}$ and $^{Ops}$) is designed to be a open source tool for the specification, assessment and optimization of QoS characteristics for cloud applications [14]. It allows users to describe a software architecture by means of MODACloudML models that express cloud-specific attributes. Its users can specify the models defining the cloud application using Creator 4Clouds graphical interface.

Space 4Clouds can be used either to assess the cost of an application and its cloud configuration or to find a suitable cloud configuration that e.g. minimizes the costs while meeting the QoS requirements, providing only the application model. Two possible usage scenarios are therefore possible. In the first one, the QoS engineer uses the tool in assessment mode, namely to evaluate the performance and cost based on a specific application deployment. In the second scenario, the QoS engineer provides only a partial configuration and lets the tool face the task of analyzing the possible alternatives to return a cost optimized solution that meets the constraints. In this scenario, the module returns a complete deployment description, and also reports useful information about the overall performance and cost. Then the QoS engineer can choose to accept the solution as it is, modify the constraints or change the deployment and evaluate again other configurations.

The feed-back loop technologies of MODAClouds Toolbox extend capabilities offered by the three main components: through them Tower 4Clouds is connected with Creator 4Clouds and Venues 4Clouds, respectively [12]. The first connector is responsible for providing the QoS engineers with the perspective of the application behavior at runtime in order to improve the development process. The second connector allows Venues 4Clouds to adapt its knowledge base according to monitoring data. This helps in offering to users an updated vision of services QoS for future recommendations.

To support QoS analysis, the developer may rely on performance, utilisation or reliability models. To provide reliable estimates, the input parameters must be accurately estimated. Such an estimation is challenging because not all QoS attributes are explicitly tracked by log files. The QoS models are initially parameterized using expert-knowledge or data collected in small deployments.

The Filling-the-Gap (FG) tool is a component for parametrization of performance models continuously at run time. Its objective is to provide accurate estimates to the parameters of the design-time QoS models [35]. Once the application has been deployed on the cloud, possibly in a production environment, the FG analysis is deployed to obtain estimates based on monitoring data collected at run-time. The FG tool implements a set of statistical estimation algorithms to parameterize performance models from run-time monitoring data related to response times and queue-lengths.

**Final remarks.** The iterative process of QoS evaluation at different layers of application description using QoS models was proved to be useful at least in the main use cases of MODAClouds project. The iteration approach can be applied also to other use cases like the ones involving cloud-based Big Data applications.

# 5 QoS Assurance for Data-Intensive Cloud Applications

Data-intensive applications, like the ones with real-time requirements, are nowadays deployed in large-scale distributed storage of cloud environments. Real-time applications, like video streaming, virtual reality, digital libraries, or scientific data collection, are rely upon a certain level of QoS for their data access needs. High-throughput data-intensive processors, like MapReduce, also needs the QoS guarantees for a fast gathering of throughput for concurrent data access. Moreover, the current techniques dealing with QoS support in distributed file systems focus mainly on the network management among heterogeneous clusters and on the flow control. How to provide storage QoS in a distributed file system in cloud environments is challenging [33].

The DICE project[4] aims at developing tools that will help software designers reasoning about efficiency, reliability and safety of data-intensive cloud applications. One of its primary goals is to provide facilities to extensively model and verify the assessment of quality aspects [5]. The continuously evaluation of the QoS is important not only in the specific verification and validation phases, but through the entire life-cycle of the services.

In this context, the recent DICE project output [6] overviews on state-of-the-art research in quality testing through model-driven approaches. Testing methods that primarily focus on quality aspects like efficiency are considered. In particular, the non-functional characteristics of the software that are of interest for DICE are reliability (including availability and fault tolerance), performance (time behavior and resource utilization) and safety (acceptable levels of risk). The main quantitative analysis techniques that are measured and compared from the point of view of assessment of performance, reliability and safety are combinatorial, state-based (stochastic Petri nets, queueing networks) or Monte Carlo simulation. For QoS assurance aspects, a list of key monitoring metrics used to assess quality are provided.

The technologies and concepts analyzed in [6] are referring to: (1) frameworks for executing data-intensive applications[5] and streaming processors[6]; (2) NoSQL databases[7] and in-memory databases[8]; (3) cloud-based blob storage[9]. Several conclusions of the analysis are reported in what follows.

**Data-intensive processors.** In what concerns the key metrics to assess the QoS of Hadoop or Spark services, the throughput and compliance with deadlines are the most relevant ones. Hadoop framework provides reliability guarantees by monitoring the execution of the map and reduce tasks. The QoS models for Hadoop reported in the literature are empirical models mapping data size into execution times, approximate formula evaluating jobs executions time starting from individual tasks execution time, or formal models like queueing network and colored Petri nets.

Big Data processing require at several stages of the computation to transfer partial results from one node to another (e.g. the copy shuffle phase of Hadoop and Spark). Data transfers could become the bottleneck of the computation. Therefore Software-Defined Networking (SDN) solutions are usually adopted to configure the underlying networking infrastructure to respond to this issue. Traffic engineering is ensuring the mechanism to optimise the performance of a data network at both traffic and resource level. SDN allows to have a centralised visibility of global network topology and status. The SDN traffic engineering mechanisms focus mainly on traffic analysis, fault tolerance, flow management, or topology update.

Stream processing solutions are designed to handle large data volumes in real time taking advantage of a highly available, scalable and fault-tolerant architecture. The key monitoring metrics are latency, non-stop streaming, I/O throughput, maximum delivery time. Streaming processors are fault-tolerant and inherently parallel, with different parts of the processors individually scalable.

**Databases.** NoSQL databases are data storage solution used nowadays for scalable web applications that have to analyze or move huge data sets. The monitoring metrics are the throughput by workload (number of operations performed on the database in the time unit, considering different types of workloads), latency by workload, and support to parallel connections. QoS attributes for NoSQL databases focuses on performance (response times, throughput, availability, scalability, consistency guarantees).

[4] www.dice-h2020.eu

[5] Current representative solutions: Hadoop and Spark

[6] Current representative solutions: Apache Storm, Apache Spark Streaming, Apache Samza, Apache S4, STReamparse, Stratio

[7] Current representative solutions: Amazon DynamoDB and SimpleDB, Google Datastore, Azure DocumentDB, Apache HBase, Cassandra, Hypertable, MongoDB, CouchDB, Riak, Redis, Berkeley DB, Dyomite

[8] Current representative solutions: Apache Derby, HyperSQL, SQLite

[9] Current representative solutions: Ceph and Amazon S3

In-memory analytics aims at processing huge volumes of data in main memory, while minimising usage of disk I/O. In their case, the key monitoring metrics include CPU utilisation, query response times, threading level, thread affinity, mean memory consumption, peak memory consumption. A significant problem for quality assurance of in-memory databases is the ability to capture time-varying threading levels that are used internally to these databases to process queries efficiently.

**Distributed block storage.** Distributed block storage (e.g. provided by Ceph and S3) is more difficult to be administrated than centralised storage, despite its enhanced scalability. In particular, tuning it for performance can be more difficult. The key monitoring metrics of Ceph are available disk space, read/write operations per second, I/O waits, network throughput. Ceph provides scalability as being able to handle thousands of client hosts accessing exabytes of data. The key S3 monitoring metrics are the size of all objects present in buckets, the number of objects present in buckets, get/push transfer speeds. The S3 service allows by design concurrent device failures as it is quickly detecting and repairing any lost redundancy.

The provision of storage QoS allowing resource reservations and allocation of the required disk bandwidth (especially requested by real-time applications or any data transfer that needs fixed bandwidth assurance) is investigated in the paper [33].

**Final remarks.** The QoS attributes used in the case of the development of cloud-based data-intensive applications are quite different from the QoS attributes were usually encountered in cloud environment (mentioned also in the first sections of this paper). Therefore the QoS assurance procedures for this special case should take into account the specific attributes.

# 6 Conclusions

Model-driven engineering techniques are proving their effectiveness in QoS assurance in cloud environment, and especially in the case of multi-clouds or data intensive application design. However the processes to be applied are quite complex and still require the intervention of human expert, a QoS engineer that is able to offer initial hints about the requested QoS of a certain application and to select, potentially after several refinement iterations, one of the recommended configurations for the application deployment on multi-cloud environments. This fact is due to the large search space and the multi-criteria objectives expressed in QoS requirements. In multi-cloud support systems the role of the QoS engineer is expected to be taken by automated procedures and the current status is showing that this requirement is not yet possible without making concessions on the QoS of the multi-cloud support system.

**Acknowledgment.**

The analysis presented in Sections 2 and 3 is partially funded by the grant of Romanian National Authority for Scientific Research, UEFISCDI, PN-II-ID-PCE-2011-3-0260 (AMICAS). Section 4 refers to the final results of the grant of European Commission FP7-ICT-2011-8-318484 (MODAClouds). Section 5 refers to the preliminary results of the grant of European Commission H2020-ICT-09-2014-644869 (DICE).

# References

[1] Ardagna, D., Casale, G., Ciavotta, M., Pérez, J.F., Wang, W.: Quality-of-service in cloud computing: modeling techniques and their applications. J. Internet Services and Applications 5:11 (2014)

[2] Ardagna, D., Ciavotta, M. (eds.): Design-time quality modelling and evaluation: analysis of the state of the art and scope definition. Deliverable D5.1 of MODAClouds. http://www.modaclouds.eu/publications/public-deliverables/ (2013)

[3] Ardagna, D., Ciavotta, M. (eds.): MODACloudML QoS abstractions and prediction models specification – initial version. Deliverable D5.2.1 of MODAClouds. http://www.modaclouds.eu/publications/public-deliverables/ (2013)

[4] Bratanis, K., Kourtesis, D.: Introducing policy-driven governance and service level failure mitigation in cloud service brokers: challenges ahead. LNCS 8377, 177–191 (2014)

[5] Casale, G., Ardagna, D., Artac, M., Barbier, F., Di Nitto, E., Henry, A., Iuhasz, G., Joubert, C., Merseguer, J., Munteanu, V. I., Pérez, J. F., Petcu, D., Rossi, M., Sheridan, C., Spais, I., Vladušič, D.: DICE: quality-driven development of data-intensive cloud applications. In Procs. MiSE '15, 78–84 (2015)

[6] Casale, G., Ustinova, T. (eds.): DICE integrated framework: state of the art analysis. Deliverable D1.1 of DICE. http://www.dice-h2020.eu/deliverables/ (2015)

[7] Chana, I., Singh, S.: Quality of service and service level agreements for cloud environments: issues and challenges. In Cloud Computing, Series Computer Communications and Networks, Springer, 51–72 (2014)

[8] Chen, J.H., Abedin, F., Chao, K.-M., Godwin, N., Li, Y., Tsai, C.-F.: A hybrid model for cloud providers and consumers to agree on QoS of cloud services. Future Generation Computer Systems 50, 38–48 (2015)

[9] Chhetri, M.B., Chichin, S., Vo, Q.B., Kowalczyk, R.: Smart CloudBenchA framework for evaluating cloud infrastructure performance. Information Systems Frontiers, 10.1007/s10796-015-9557-2 (2015)

[10] Chu, W.C.-C., Yang, C.T., Lu, C.-W., Chang, C.-H., Hsueh, N.-L., Hsu, T.-S., S.H.: An Approach of quality of service assurance for enterprise cloud computing (QoSAECC), In Procs. TSA, 7–13 (2014)

[11] CSMIC: Service Measurement Index. Framework version 2.1. http://csmic.org/ downloads/SMI_Overview_TwoPointOne.pdf (2014)

[12] Di Nitto, E., Matthews, P., Petcu, D., Solberg, A.: Model-driven development and operation of multi-cloud applications. The MODAClouds approach. SpringBriefs series, Springer (to appear)

[13] Domínguez-Mayo F.J., García-García, J.A., Escalona, M.J., Mejías, M., Urbieta, M., Rossi, G.: A framework and tool to manage Cloud Computing service quality, Software Qual J, 10.1007/s11219-014-9248-0 (2014)

[14] Franceschelli, D., Ardagna, D., Ciavotta, M., Di Nitto, E: Space4cloud: a tool for system performance and cost evaluation of cloud systems. In Procs. MultiCloud'13, 27–34 (2013)

[15] Hershey, P.C., Rao, S., Silio, C.B., Narayan, A.: System of systems for quality-of-service observation and response in cloud computing environments. IEEE Systems J. 9 (1), 212–222 (2015)

[16] International Telecommunication Union, E.800 : Definitions of terms related to quality of service. http://www.itu.int/rec/T-REC-E.800-200809-I (2008)

[17] Karawash, A., Mcheick, H., Dbouk, M.: Quality-of-service data warehouse for the selection of cloud services: a recent trend. In Cloud Computing, Series Computer Communications and Networks, Springer, 257–276 (2014)

[18] Kourtesis, D., Alvarez-Rodríguez, J.M., Paraskakis, I.: Semantic-based QoS management in cloud systems: Current status and future challenges. Future Generation Computer Systems 32, 307–323 (2014)

[19] Kourtesis, D., Bratanis, K., Friesen, A., Verginadis, Y., Simons, A.J. H., Rossini, A., Schwichtenberg, A., Gouvas, P.: Brokerage for quality assurance and optimisation of cloud services: an analysis of key requirements. LNCS 8377, 150–162 (2014)

[20] Lee, S.-Y., Tang, D., Chen, T., Chu, W.C.-C.: A QoS assurance middleware model for enterprise cloud computing. In Procs. IEEE 36th COMPSACW, 322–327 (2012)

[21] Lin, J.-W., Chen, C.-H., Lin, C.-Y.: Integrating QoS awareness with virtualization in cloud computing systems for delay-sensitive applications. Future Generation Computer Systems 37, 478–487 (2014)

[22] Maarouf, A., Marzouk, A., Haqiq, A.: Automatic control of the quality of service contract by a third party in the cloud computing. Procs. WCCS, 599–603 (2014)

[23] Mao, C., Chen, J., Towey, D., Chen, J., Xie, X.: Search-based QoS ranking prediction for web services in cloud environments. Future Generation Computer Systems 50, 111–126 (2015)

[24] McIntyre, S.H., Kirby, G.: The market for service assurance: a model and data structure. In Procs. SRII Global Conference, 184–192 (2011)

[25] Petcu, D.: Consuming resources and services from multiple clouds. From terminology to cloudware support. J. Grid Computing 12 (2), 321–345 (2014)

[26] Petcu, D., Vasilakos, A.V.: Portability in clouds: approaches and research opportunities. Scalable Computing: Practice and Experience 15 (3), 251–270 (2014)

[27] Quarati, A., Clematis, A., D'Agostino, D.: Delivering cloud services with QoS requirements: business opportunities, architectural solutions and energy-saving aspects. Future Generation Computer Systems, 10.1016/j.future.2015.02.009 (2015)

[28] Rak, M., Suri, N., Luna, J., Petcu, D., Casola, V., Villano, U.: Security as a service using an SLA-based approach via SPECS. In Procs. 5th CloudCom 2, 1–6 (2013)

[29] Rao, J., Wei, Y., Gong, J., Xu, C.-Z.: QoS guarantees and service differentiation for dynamic cloud applications. IEEE Transactions on Network and Service Management 10 (1), 43–55 (2013)

[30] Thoss, Y., Pohl, C., Hoffmann, M., Spillner, J., Schill, A.: User-friendly visualization of cloud quality. In Procs. IEEE 7th CLOUD, 890–897 (2014)

[31] Veloudis, S., Friesen, A., Paraskakis, I., Verginadis, Y., Patiniotakis, I.: Underpinning a cloud brokerage service framework for quality assurance and optimization. In Procs. IEEE 6th CloudCom, 660–663 (2014)

[32] Veloudis, S., Paraskakis, I., Friesen, A., Verginadis, Y., Patiniotakis, I., Rossini, A.: Continuous quality assurance and optimisation in cloud-based virtual enterprises. In Collaborative Systems for Smart Networked Environments, Series IFIP Advances in Information and Communication Technology 434, 621–632 (2014)

[33] Wang, C.-M., Yeh, T.-C., Tseng, G.-F.: Provision of storage QoS in distributed file systems for clouds. In Procs. 41st ICPP, 189–198 (2012)

[34] Wang, S., Liu, Z. Sun, O., Zou, H., Yang, F.: Towards an accurate evaluation of quality of cloud service in service-oriented cloud computing. J. Intelligent Manufacturing 25, 283–291 (2014)

[35] Wang, W., Perez, J. F., Casale, G.: Filling the Gap: A Tool to Automate Parameter Estimation for Software Performance Models. In Procs. 1st QUDOS, 31–32 (2015)

[36] Wollersheim, J., Krcmar, H.: Quality analysis approaches for cloud services - towards a framework along the customer's activity cycle. In Trusted Cloud Computing, Springer, 109–124 (2014)