Liet. mat. rink. LMD darbai, **50**, 2009, 357–361

Conceptual analysis of the notion of quality of services*

Jérémy BESSON, Albertas ČAPLINSKAS
Institute of Mathematics and Informatics
Akademijos 4, LT-08663 Vilnius, Lithuania
e-mail: contact.jeremy.besson@gmail.com; alcapl@ktl.mii.lt

Abstract. In the last decade the component technologies have evolved from object-oriented to service-oriented ones. Services are seen as utilities based on a pay-for-use model. This model requires providing and guaranteeing a certain Quality of Service (QoS). However, QoS and even a service itself can be defined and understood in many different ways. It is by far not obvious which of these approaches and in what extent they should be used when developing service-oriented software systems. This paper analyzes the notion of QoS namely from this point of view.

Keywords: quality of services, component systems, service-oriented approach.

Introduction

Software Engineering community faces a period of revolutionary changes [3]. In the last decade the component technologies have evolved from object-oriented to service-oriented ones. In this context, the one of most important concepts is Quality of Service (QoS). However, the term QoS can be defined in many different ways. Analysis and understanding of various interpretations of Quality of Service is important not only for theoretical reasons but also for highly practical ones because this determines how one should to conceptualize and formalize QoS when developing service-oriented software systems.

1. What is a service?

Mainly, a service is an *utility* based on a pay-for-use model. However, the precise definition of this concept is still not agreed upon. It depends also on the point of view. For example, from business point of view, a service is a common activity, which, when applied under various contexts over different business entities, results in a business circumstance that is uniquely distinguishable. But from technological point of view it may be defined as a self contained, replaceable and reusable software component that exhibits high cohesion of functional/semantic relatedness of activities and is loosely coupled through multiple standard interfaces and bindings. In the context of emerging Internet of Services (IoS), many researchers thought of services as an Internet-based version of traditional business services. Some authors [7,9] see the Internet only as a

*The research is funded by the Lithuanian State Science and Studies Foundation under the contract No. S 09/2009.

channel to interact with customers or as a kind of user interface to access services. For others [10] a service in the context of IoS i s a provision of service over e-networks, where e-networks are as wide as the traditional Internet and include wireless networks and various e-environments such as ATMs, smart card networks, etc. The detailed analysis of service-related terms from business science, information science and computer science perspectives has been done in [1]. An attempt to standardise this concept has been done by the TM Forum's Information Framework (SID) [4]. SID defines a product as what the enterprise sells or delivers; *services* as those things that create, deliver or support a product; and *resources* as what comprises a service. SID differs between a *customer-facing service* (CFS) and a *resource-facing service* (RFS). CFS is an abstraction defining the characteristics and behaviour of a particular service as seen by the customer who acquires, purchases and/or is directly aware of the type of service. RFSs support CFSs but are not seen or purchased by the customer. Such services are used to build CFSs. S o, CFS consists of RFSs, and, in turn, RFS consists of resources, such as software, servers, routers, individual network links, etc.

There exist a great variety of e-services. Apart well-known web-services, the list includes client-server services (e.g., e-mail, FTP, web browsing), multimedia conversational services (e.g., video, audio, data conferencing), playback services (e.g., radio webcasting, TV webcasting), computing services, hardware as a service (e.g., cloud computing, cloud storage, backup as a service), database as a service, platform as a service (e.g., user interface as a Service, Google App Engine, Etelos, QuickBase), content as a service, data as a service (e.g., Salary.com), data mining as a service, desktop as a service, security services, etc. Up to date, there does not exist any exhaustive taxonomy of e-services.

2. Quality of services

The most general definition of the QoS is an agreed or contracted level of service between a service customer and service provider. However, this definition cannot be used directly to evaluate or maintain the QoS of component systems.

Initially, the concept of QoS has been developed for the telephony and other network services. Later, it has been extended to be applicable to IP-networks and even to heterogeneous wireless environments. The IP was designed to provide best-effort service for delivery of data packets. However, to manage the variety of applications such as streaming video, voice over IP or e-commerce a network requires QoS in addition to best-effort service. For video and audio media and other continuous media it is usually required to guarantee some acceptable levels of service. Besides, in new emerging environment communication requirements are extremely diverse and application-dependent. For example, critical distributed applications require that the QoS would guarantee both reliability and bounds on message latency. To facilitate true end-to-end QoS on an IP-network, the Internet Engineering Task Force (IETF) has defined two models – Integrated Services (IntServ) and Differentiated Services (DiffServ). IntServ provides a rich end-to-end QoS solution, using end-to-end signalling, state-maintenance and admission control at each network element. This model relies on the Resource Reservation Protocol (RSVP) to signal and reserve the desired QoS for each flow in the network. In IntServ model every device along the path of a packet

must be fully aware of RSVP and capable of signalling the required QoS. The main drawback of this approach is that state information for each reservation must be maintained at every device along the path. DiffServ provides relatively simple and coarse methods of categorizing traffic into different classes, classes of services, and applies QoS parameters to those classes. However, in both architectures the notion of QoS is extremely narrow, because they both are based on a best effort performance model. Other limitation is the static nature of service provision. It means that the value of a QoS parameter remains the same through the lifetime of a connection an d can never be re-negotiated. If a service provider is unable to maintain its commitment there is no way to inform the service consumer and allow him to negotiate a lower QoS. Heterogeneous wireless networks support the coexistence of a variety of wireless access technologies and a number of applications and services with different QoS requirements. Moving from one access network to another, a customer might interact with different network capacity, channel characteristics, QoS management mechanisms and policies. It means that the QoS should hide the variety of lower layers. One of the main differences of cellular mobile networks compared to wireline ones is a handoff phenomenon. A mobile terminal can change its point of attachment to the network and this can lead to a resource shortage because the negotiated bandwidth is no longer available after a handoff. It means that the handoff success probability must be considered as a mobility-specific QoS parameter. Consequently, a specific s ervice model for heterogeneous wireless networks is needed. One of the most promising candidates is the Universal Mobile Telecommunication System (UMTS) bearer service layered architecture [8]. In this architecture, each bearer service on a specific layer offers its individual services using services provided by the layers below. Like DiffServ, this architecture provides the categorisation of services into different classes according to QoS requirements. UMTS defines four QoS traffic classes: conversational, streaming, interactive, and background where background class is similar to the best effort traffic. However, the end-to-end QoS, in which customers are interested, depends not only on the quality of networks but also on the quality of other resources (i.e., servers, components, execution environments) and on the quality of RFS. Besides, many CFS are compositional ones. The number of transactions which must be monitored, recorded and processed increases dramatically when complex ser vices are delivered across a far longer and multi-dimensional value chain. Thus, to predict and maintain the end-to-end QoS is very hard problem that still is unsolved. The evaluation of the quality of software components which generate required services and the prediction of the behaviour of end hosts on which these components residue is an integral part of this problem. As stated in [12], “an application that does not or cannot cooperate with the network is liable to behave in such a way as to negate the network’s efforts”.

In component-based software engineering (CBSE) the QoS specifications were developed to take the design decisions during the system development phase. In this case, the QoS requirements (more exactly, non-functional requirements) should be met statically by proper design and configuration choices. The ISO 9126 standard [5] defines a number of QoS characteristics for software components. A characteristic represents some aspect of the QoS of a service that can be identified and quantified. QoS characteristics are quantified with some specific parameters and methods, and

with other characteristics of lower abstraction level. They can be grouped into categories that group characteristics of a common subject. Different consumers of service use the same characteristics, but they use different evaluation methods or establish different hierarchies. QoS constraints enable to express limitations in the parameters and methods of characteristics. However, an agreed upon and exhaustive list of QoS characteristics still does not exist. Besides, the services are usually delivered in Open Distributed Processing (ODP) environment and some characteristics of QoS depend on the characteristics of this environment or, in other words, sometimes QoS can be a function of how the environment performs. To provide QoS mechanisms suitable for this aim an international standard has been developed [6]. The ODP middleware (e.g., CORBA, J2EE, Microsoft.NET) raises the level of abstraction but many design flaws still cannot be discovered until the components' integration time. It is impossible to perform any preliminary QoS provision of the entire system. For example, an overloaded Web server can have such an impact on user perceived response time that this time can becomes completely unacceptable for the user. Thus, new techniques should be developed to predict and evaluate QoS for large ODP systems. Additional problems arise in the Grid environment where a number of non-dedicated hosts exist and where each network is composed of several homogeneous or heterogeneous computational resources. In this case, we have a list of applications whose arriving rates follow the Poisson distribution and each application has its own specific QoS request.

3. Conclusions

Service engineering is an emerging discipline. There still exist a lot of open questions and only some preliminary answers. What is a service? What is the difference between a service and an e-service? What taxonomy should be used to classify Internet-based services? What does really mean the QoS? How to map the quality requirements from the higher levels to lower ones? How to evaluate the QoS of compositional services? It is not an exhaustive list of research questions which should be answered in order to build the Internet of Services. It seems that new mathematical models must be developed to serve as a theoretical basis for service engineering. The current operation research theory, including the traditional queuing theory, cannot cope well with many services and operation problems because of the sociotechnical nature of service-oriented systems. For example, such QoS characteristics as thrust, easy of use, security and cultural affinity should be formulated and modelled using qual itative approaches. Using typical assumptions made in queuing theory models, it is impossible to predict the performance among the various servers of N-tier applications because of their interdependencies and asymmetry. Therefore, service-based operations research and management is in demand [2,11].

Another big challenge is to guarantee QoS levels in large-scale open environments that consist of interacting software components generating cascades of services, various execution environments, servers, communication systems, networks and other resources. Up to time, many isolated areas of QoS provision, especially, communication systems, networks and HTTP servers [5] have been investigated in detail. However, little attention has been paid to the development of a coherent integrated framework defining QoS mechanisms across all architectural layers.

References

1. Z. Baida, J. Gordijn, B. Omelayenko. A shared service terminology for online service provisioning. In: M. Janssen, H.G. Sol, R.W. Wagenaar (Eds.), *Proc. of the 6th Intern. Conference on Electronic Commerce (ICEC'04)*, Delft, The Netherlands, ACM International Conference Proceeding Series 60, ACM Press, 1–10, 2004.
2. P. Bell. Operations research for everyone (including poets). *OR/MS Today*, 32(4):22–27, 2005.
3. J. Domingue, D. Fensel, R. GonzAlez-Cabero. SOA4All, enabling the SOA revolution on a world wide scale. In: *Proc. of 2008 IEEE International Conference on Semantic Computing*, IEEE Computer Society, 530–537, 2008.
4. *Information Framework (SID)*. 1988-2009, TeleManagement Forum. http://www.tmforum.org/InformationFramework/1684/home.html?catID=1684
5. ISO/IEC 9126. *Information Technology – Software Product Evaluation – Quality Characteristics and Guidelines for their Use*, first edition, 1991-12-15, reference number ISO/IEC 9126: 1991(E).
6. ISO/IEC CD 10746-1. *Information technology – Open Distributed Processing – Reference Model: Overview*, first edition, 1998-12-15, reference number ISO/IEC 10746-1:1998(E).
7. S. Janda, P.J. Trocchia, K.P. Gwinner. Consumer perceptions of internet retail service quality. *International Journal of Service Industry Management*, 13(5):412–431, 2002.
8. J. Laukkanen. *Quality of Service Concept and Architecture*. Advanced Wireless Communication Systems, Department of Computer Science, University of Helsinki, Helsinki, 2000.
9. A.C.R. van Riel, V. Liljander, P. Jurriëns. Exploring consumer evaluations of e-services: a portal site. *International Journal of Service Industry Management*, 12(4):359–377, 2001.
10. R.T. Rust, P. Kannan. E-service: a new paradigm for business in the electronic environment. *Communications of the ACM*, 46(6):36–42, 2003.
11. R.G. Qui. *Enterprise Service Computing: From Concept To Deployment*. Idea Group Pub, 2007.
12. N. Vasiliou. *Overview of Internet QoS and Web Server QoS*. Reading Course Paper, Computer Science Dept., University of Western Ontario, Canada, 2000.

REZIUMĖ

J. Besson, A. Čaplinskas. Paslaugos kokybės sąvokos koncepcinė analizė

Pastarąjį dešimtmetį objektinės komponentinės technologijos peraugo į paslaugų teikmo technologijas. Paslaugos traktuojamos kaip apmokami viešojo pobūdžio patarnavimai. Tokia paslaugų traktuotė reikalauja planuoti ir užtikrinti teikiamų paslaugų kokybę. Tačiau paslaugų kokybė gali būti suprantama ir apibrėžiama gana skirtingai. Neakivaizdu, kuria iš šių sampratų ir kokiu mastu tikslinga naudotis, kuriant paslaugoms teikti pritaikytas komponentines sistemas. Straipsnyje bandoma atsakyti į šiuos klausimus, atliekant paslaugų kokybės sąvokos koncepcinę analizę.

Raktiniai žodžiai: komponentinės sistemos, paslaugoms teikti pritaikytos sistemos, paslaugų kokybė.