QoS Analysis for Web Service Applications: a Survey of Performance-oriented Approaches from an Architectural Viewpoint

Moreno Marzolla **Raffaela Mirandola**

Technical Report UBLCS-2010-05

February 2010

Department of Computer Science
University of Bologna
Mura Anteo Zamboni 7
40127 Bologna (Italy)

The University of Bologna Department of Computer Science Research Technical Reports are available in PDF and gzipped PostScript formats via anonymous FTP from the area `ftp.cs.unibo.it:/pub/TR/UBLCS` or via WWW at URL `http://www.cs.unibo.it/`. Plain-text abstracts organized by year are available in the directory `ABSTRACTS`.

Recent Titles from the UBLCS Technical Report Series

2008-17 *Measures of conflict and power in strategic settings*, Rossi, G., October 2008.

2008-18 *Lebesgue's Dominated Convergence Theorem in Bishop's Style*, Sacerdoti Coen, C., Zoli, E., November 2008.

2009-01 *A Note on Basic Implication*, Guidi, F., January 2009.

2009-02 *Algorithms for network design and routing problems (Ph.D. Thesis)*, Bartolini, E., February 2009.

2009-03 *Design and Performance Evaluation of Network on-Chip Communication Protocols and Architectures (Ph.D. Thesis)*, Concer, N., February 2009.

2009-04 *Kernel Methods for Tree Structured Data (Ph.D. Thesis)*, Da San Martino, G., February 2009.

2009-05 *Expressiveness of Concurrent Languages (Ph.D. Thesis)*, di Giusto, C., February 2009.

2009-06 *EXAM-S: an Analysis tool for Multi-Domain Policy Sets (Ph.D. Thesis)*, Ferrini, R., February 2009.

2009-07 *Self-Organizing Mechanisms for Task Allocation in a Knowledge-Based Economy (Ph.D. Thesis)*, Marcozzi, A., February 2009.

2009-08 *3-Dimensional Protein Reconstruction from Contact Maps: Complexity and Experimental Results (Ph.D. Thesis)*, Medri, F., February 2009.

2009-09 *A core calculus for the analysis and implementation of biologically inspired languages (Ph.D. Thesis)*, Versari, C., February 2009.

2009-10 *Probabilistic Data Integration*, Magnani, M., Montesi, D., March 2009.

2009-11 *Equilibrium Selection via Strategy Restriction in Multi-Stage Congestion Games for Real-time Streaming*, Rossi, G., Ferretti, S., D'Angelo, G., April 2009.

2009-12 *Natural deduction environment for Matita*, C. Sacerdoti Coen, E. Tassi, June 2009.

2009-13 *Hints in Unification*, Asperti, A., Ricciotti, W., Sacerdoti Coen, C., Tassi, E., June 2009.

2009-14 *A New Type for Tactics*, Asperti, A., Ricciotti, W., Sacerdoti Coen, C., Tassi, E., June 2009.

2009-15 *The k-Lattice: Decidability Boundaries for Qualitative Analysis in Biological Languages*, Delzanno, G., Di Giusto, C., Gabbrielli, M., Laneve, C., Zavattaro, G., June 2009.

2009-16 *Landau's "Grundlagen der Analysis" from Automath to lambda-delta*, Guidi, F., September 2009.

2010-01 *Fast overlapping of protein contact maps by alignment of eigenvectors*, Di Lena, P., Fariselli, P., Margara, L., Vassura, M., Casadio, R., January 2010.

2010-02 *Optimized Training of Support Vector Machines on the Cell Processor*, Marzolla, M., February 2010.

2010-03 *Modeling Self-Organizing, Faulty Peer-to-Peer Systems as Complex Networks* Ferretti, S., February 2010.

2010-04 *The qnetworks Toolbox: A Software Package for Queueing Networks Analysis*, Marzolla, M., February 2010.

QoS Analysis for Web Service Applications: a Survey of Performance-oriented Approaches from an Architectural Viewpoint

Moreno Marzolla[1] **Raffaela Mirandola**[2]

Technical Report UBLCS-2010-05

February 2010

Abstract

Building applications from independently developed services is one of the current trends in software development. In this context, the quality–and specifically performance–of a composition of services is a crucial factor for deciding which components must be selected, or to choose whether a given sequence of interactions can provide the requested quality of service. To achieve this goal, different methods and tools to capture and analyze the performance of web services have been developed. These methods and tools aim at helping Web service providers and consumers to understand design trade-offs, optimize the system by identifying performance bottlenecks, or analyze the system performance within a specified deployment environment.

The goal of this paper to propose an initial taxonomy for analyzing the existing performance prediction and analysis methods for the development of service-based systems. Then, we use this taxonomy to discuss the strengths and weaknesses of different performance prediction approaches, trying to establish a basis to select an appropriate prediction method and to provide recommendations for future research activities.

1. Department of Computer Science, University of Bologna, Mura Anteo Zamboni 7, I-40127 Bologna, Italy, Email: marzolla@cs.unibo.it
2. Dipartimento di Elettronica e Informazione, Politecnico di Milano, via Ponzio 34/5, 20133 Milano, Italy, Email: mirandola@elet.polimi.it

1 Introduction

During recent years, the way software systems are designed and built is undergoing great changes. One of the major current trends is the development of applications created from independently developed services. In this vision, providers offer similar competing services corresponding to a functional description of a service; these offerings can differ significantly in some Quality of Service (QoS) attributes like performance [20, 21]. On the other side, prospective users of services dynamically choose the best offerings for their purposes. Using the Service Oriented Architecture (SOA) paradigm to build applications, services can be dynamically selected and integrated at run-time, so enabling system properties like flexibility, adaptiveness, and reusability.

In this context, the quality–and specifically performance–of a composition of services is a crucial factor for deciding which components must be selected, or to choose whether a given sequence of interactions can provide the requested QoS. However, while managing QoS in distributed systems is not a novel problem, a number of additional issues arise in the context of a service-oriented computing environment, due to the nature of these applications. In fact, service based applications are built by integrating pieces of software (possibly developed by different stakeholders) in a distributed setting and the configuration can change during run time. Besides, applications built on services should ensure, on one hand, that users experience the required performance and on the other hand that provider incomes are maximized. Due to the high dynamism of the applications, the quality assessment should be performed both at development and at run time. In fact the quality of the application depends not only on the selected services but also on the underlying support systems and on the network resources.

To achieve this goal, different methods and tools to capture and analyze the performance of Web Service (WS) have been developed. In general, the proposed approaches for evaluation of quality of service capabilities for WS are quite different each from the other. Each one focuses on a different set of QoS metrics and can be applied at run-time or system design time. Also, the existing approaches are built upon different QoS models and techniques. In this scenario, it can be difficult to choose the right approach which can be applied to a specific QoS evaluation study.

In this paper we want to address this issue. After a first analysis of the main QoS requirements, we focus on the performance attribute and we propose an initial taxonomy to analyze the various performance prediction and analysis methods for the development of service-based systems. Then, we use this taxonomy to discuss their respective strengths and weaknesses. In doing so, we establish a basis to select an appropriate prediction method and to provide recommendations for future research activities.

We observe that the literature of performance approaches for WS is extremely large. To made things worse, in most cases the general performance evaluation and estimation methodologies can be applied to the WS world with slight modifications, as well. That would further increase the size and complexity of our analysis. In this paper we concentrate on architectural-level performance evaluation of WS applications; for this reason, at the moment we do not consider most of the QoS evaluation approaches based on system and services monitoring, as monitoring requires (by definition) a running system to be monitored, and thus cannot be applied at the architectural design level. However, this work represents only an initial step towards a definition for a taxonomy of QoS measurement and prediction for WS, and can constitutes a starting point for further, broader investigations.

The paper is organized as follows. In Section 2 we give an overview of the QoS requirements and their meaning in the context of WS. In Section 3 we describe the attributes we use to classify the QoS approaches. In Section 4 we describe some recent QoS prediction and measurement approaches according the the previously illustrated attributes. We summarize the classification in Section 5 and we present the conclusions in Section 6.

2 QoS Requirements

Since 2002 Menasce [20] highlighted the need for QoS definition, specification and evaluation in WS from the perspective of both service provider and service user. In 2003 the W3C [1] sum-

marized the key requirements of QoS for WS. A good review paper on the state-of-art in the field has been presented by Ludwig in [16]. Recently, several research works dealt with the definition of QoS languages for WS-based applications. This includes HP's Web Services Management Language (WSML) and framework, IBM's Web Service Level Agreement (WSLA) language, the Web Services Offer Language (WSOL) as well as approaches based on WS-Policy[16].

There are many possible QoS metrics which can be considered; however, the metrics which are usually relevant for WS are the following [1]:

Performance The performance of a web service represents how fast a service request can be completed. Depending on the stakeholders interests several figures of merit can be defined to deal with performance, such as throughput, response time, execution time and resource utilization. *Throughput* is the number of completions of web service requests during an observation time interval. *Response time* the period of time necessary to complete a web service request. *Execution time* is the resource time consumed by a web service to process its composing activities. *Resource utilization* is the percentage of time a resource is busy serving web services activities [13].

Dependability The dependability of WSes is a property that integrates several attributes, the most important of which are: *Reliability* (representing the ability of a WS to perform its required functions under stated conditions for a specified time interval), *Availability* (i.e., the probability that the system is up) and security.

Price WS can be provided against payment of a certain amount of money; usually the *price* of a WS is defined by the provider. The price may be fixed for each invocation, or proportional to the actual (measured) service demand to each WS method that is used. Providers may also request higher fees for services hosted on better/faster hardware.

Reputation The WS reputation reflects a common perception of other WS or customer towards that service. In other words, it aggregates the ratings of the given service by other principals. Typically, a reputation would be built from a history of ratings by various parties [17]. Several new models and metrics have been defined to deal with this new attribute (interested reader can refer to the conference series ICWS [3] and ICSOC [2]), however in this paper we don't tackle this issue but it remains object of future work.

3 A taxonomy of concerns

We now describe a set of attributes (see Fig. 1) which will be used to classify each approach. These will form the basis for the discussion carried out in Section 5.

Point of View We consider two different point of views: *provider* and *consumer* Point Of View (PoV). Each approach may evaluate the QoS from the service provider o consumer PoV (or both). This distinction is meaningful as provider and consumer have usually very different–even anthitetic–goals. The provider is interested in maximize the resource utilization or the economic profits while providing an adequate service level. The consumer is interested in minimizing the cost, while maximizing other QoS metrics such as reliability or throughput.

Development Level Execution of a distributed application which consists of the invocation of different WS is a complex task, which can be done only after the needed WS have been located, selected and integrated. QoS attributes can be managed both at *Run-Time* or ad *Design Time*: quality assessment should not be deferred at the end of the development phase, but rather incorporated in the selection and integration activities.

The quality assessment should be performed both at development and at run time. In fact the quality of the application not only depends on the selected services but also on the underlying support systems and on the network (resources).

QoS metric Different QoS parameters require different types of resources and different management approaches. *Performance management* from a provider point of view deals with shared or dedicated resources to be allocated to a particular scope of quality. The service providers are mainly concerned about utilization and throughput of their resources; this is because they usually want to ensure maximum resource utilization and maximum throughput so that they are able to serve the largest number of customers. On the other hand, from the consumer side performance

management is mainly related to the observed response time. *Reliability* is an important QoS metric, especially in the context of distributed Web applications. *Cost* is another important QoS metric. Providers want to maximize their income, while customers want to obtain an adequate service level (in term of performance, reliability, security) with the minimum cost.

QoS enhancement methods Different methods and tools can be used to predict/analyze QoS of WS, we outline the main approaches followed so far in the literature:

Models Stochastic models are widely used in the performance and reliability evaluation of computer and communication systems. Different kinds of models have been proposed including single queueing systems, queueing networks, timed stochastic Petri nets, Markov and non-Markov stochastic processes. Various types of model can be applied to represent specific characteristics of classes of systems, such as queueing networks for congestion systems and timed stochastic Petri nets for concurrent systems, Markov model to study reliability and availability. Models can be classified in two main categories: analytical and simulation models depending on the analysis approach. An analytical model of a system is represented by a set of variables and parameters to represent system components and a set of equations that correspond to their interactions.

Ontology This term became popular in the last years to indicate a (specification of a) conceptualization of a knowledge domain, qualities of services in our case. Roughly speaking an ontology can be seen as a taxonomy with some additional features such as semantic relationships among terms and attributes or well defined rules about the way to define terms and relationships. The definition of a QoS ontology provides a basis for providers to advertise their offerings, for consumers to express their preferences, and for ratings of services to be gathered and shared. In such a way, the matching between demands and offerings can be done semantically and dynamically. The semantic matching allows the service agent to match consumers to services using the providers advertised QoS policy for the services and the consumers QoS preferences. The provider policy and consumer preferences are expressed using the concepts in the ontology.

Measurement and Monitoring Monitoring is a runtime activity whose objective is to collect data about a phenomenon and analyze them with respect to certain criteria. The data to be collected and the analysis criteria must be defined as part of the monitoring parameters, and may change dynamically. The main goal of monitoring activities is to discover potential critical problems while a system is executing. In general terms, the monitoring system is meant to perform continuous monitoring of a set of services, by analyzing relevant extra-functional properties; in addition it can also check the behavior of invoked services against a certain specification. In real-life scenarios, a naive approach to monitoring can introduce considerable QoS degradation, influencing those very parameters that one aims to measure.

Selection or Composition In the SOA context, the key point is to build applications through the selection and/or composition of available services. By *service selection* we denote the activity of choosing, among a number of alternatives, the services providing the required functionalities. It may be possible that different implementations providing the exact same functionality, with different QoS, are available. In this case, it is necessary to select the "best" alternative, depending on some optimality criteria (i.e., select the service with minimum cost, or with better response time). By *service composition* we denote the definition of an integration schema yielding the target application by means of composition of available services, each one providing a simple functionality. Again, given a set of basic building blocks, the same complex behavior may be obtained by different composition schemas of different blocks; it is thus necessary to select the "best" composition schema, according to some optimality criteria.

Current SOA approaches only partially address this global vision. While services are described and listed in public registries, there is little or no support for actually making quality-based service selection and integration. Therefore, QoS support for WS has recently become a very active area of research and standardization, involving major challenges such as QoS-aware service description, composition, and selection (e.g., [8, 20]).

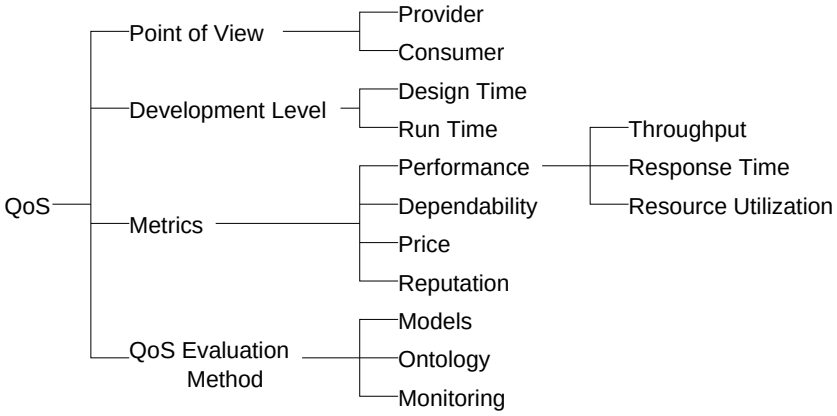

Figure 1. The Taxonomy used in this paper

4 Classification of approaches

Quality of Service (QoS) and specifically performance has been extensively studied in the context of computer systems and networks. However, performance in the context of software engineering and WS has seen a flurry of recent research activity. Due to space limitation, in the following we restrict our attention to relevant literature on resource optimization, QoS-driven service discovery and selection, QoS models and metrics and QoS-aware frameworks, while in the next section these works are compared/classified taking into account the concerns defined in Section 3.

4.1 Resource optimization

Service Level Agreement (SLA) defines a set of consumer expectations which must be met by a provider if a contract is not to be broken. Since providers will potentially be offering many different services to different consumers, they must adopt an efficient policy for resource management which differentiates consumers into service ranges. To this end several works exist in the literature that focus on resource management by improving the QoS elements like response time and throughput and, in some cases, the reliability of the servers(see [16, 26] and references therein for an overview). Classical approaches based on optimal resource allocation have been applied [11], or ad-hoc methods have been developed based on differential resource allocation schemes for different requests based on the request type [14]. These kinds of resource allocation have been made on the basis of either request content, or the percentages of request arrivals. Other approaches focus on building infrastructure that controls the order (schedule) in which the requests are picked up for execution by a resource to effect [14]. These methods are based on QoS monitoring systems. In [26] the authors evaluate different approaches when providers adopt a policy based on service differentiation and in response introduce and evaluate an expectation-based approach to QoS assessment which presupposes the classification of consumers into ranges defined by their expectation. As well as carrying out assessment to determine the likely future behavior of a provider for a given consumer expectation, a confidence value is attached to this assessment to indicate the level of certainty that the result is accurate. The obtained results suggest that this confidence-based approach can help consumers make better informed decisions in order to find the providers that best meet their needs. In [28] the authors describe a resource allocation algorithm used by a QoS brokering service. The goal of the resource allocation strategy is to maximize the resource utilization at the service provider side while trying to meet the user-defined QoS requirements. Panzieri et al. [15] describe a QoS-aware middleware which can be include in application servers in order to implement SLA-driven clustering of servers. The middleware operates by dynamically configuring, monitoring and balancing the load among different servers, in order to provide strong QoS guarantees despite high variability in the request rates.

4.2 QoS-driven service discovery and selection

Different approaches have been followed so far, spanning the use of QoS ontology, the definition of ad-hoc methods in some general framework, and the exploitation of optimization algorithms.

Considering the use of ontologies, Maximilien and Sing [18] address dynamic service selection via an agent framework coupled with a QoS ontology. With this approach, participants can collaborate to determine each others service quality and trustworthiness. The same authors describe in [19] a service selection process based on a trust model taking into account a shared QoS conceptualization that considers the preferences for qualities when determining the trust value to assign to service instances. Vu et al. [27] present a QoS-based semantic WS selection and ranking solution with the application of a trust and reputation management method. A quite different approach is proposed by Casati et al. [9], that present a dynamic service selection using a data mining approach on the service conversation logs. that can dynamically analyze the logs of various conversations and determine the services that best satisfy the service consumer's goals [9].

Other works concern different kinds of optimization algorithms for the selection of concrete services in a composite service [4, 5, 10, 30, 31]. Yu and Lin [30] discuss selection algorithms for multiple QoS attributes defining the problem as a multi-dimension multi-choice 0-1 knapsack one as well as a multi-constraint optimal path problem. Zeng et al. [31] present a global planning approach to select an optimal execution plan by means of integer programming. They propose a simple QoS model using the attributes: price, availability, reliability, and reputation. They apply linear programming for solving the optimization QoS matrix formed by all of the possible execution plans to obtain the maximum QoS values. Ardagna and Pernici [4] model the service composition as a mixed integer linear problem where both local and global constraints are taken into account. Their approach is formulated as an optimization problem handling the whole application instead of each execution path separately. Claro et al. [10] propose the use of multi-objective optimization techniques to find a set of optimal Pareto solutions from which a requestor can choose. Canfora et al. [5] adopt a quite different strategy for optimal selection based on genetic algorithms. An iterative procedure is defined to search for the best solution of a given problem among a constant size population without the need for linearization required by integer programming.

The approach presented in [6] differs from previous works which have tackled the service selection as an optimization problem in that the optimization is performed on a per-flow rather than per-request basis. In case of high volumes of service requests, per-request service selection approaches may suffer from scalability problems because of the computational overhead for solving the optimization problem for each single requests (also more times per request, according to some proposal [31]). On the contrary, in this approach the solution of the optimization problem holds for all the requests in a flow, and is recalculated only when some significant event occurs (e.g., a change in the availability or the QoS values of the selected concrete services). Moreover, the broker solves the optimization problem taking into account simultaneously the flows of requests generated by multiple requestors, with possibly different QoS constraints.

4.3 QoS of WS composition and QoS-aware frameworks

While in some cases WS may be utilized in an isolated form, it is natural to expect that WS will be integrated as part of workflows. In this section we describe some of the approaches which have been proposed in the literature to evaluate QoS characteristics of workflows.

Cardoso et al. [7] propose an approach for evaluating different QoS characteristics of WS processes by means of a technique called Stochastic Workflow Reduction. This can be applied to estimate different QoS attributes, such as cost, reliability and response time. Different QoS measures for whole workflows based on atomic QoS attributes by means of graph reductions on the flow-graph describing the workflow.

The paper [12] proposes a mechanism that implements an optimizing WS composition combining performance optimization, price optimization, and payload optimization when meeting the requirements of SLA. The paper defines a negotiation arithmetic, which is applied to the set of the business requirements and the vector space of the relative parameters of candidate WS. Af-

ter analyzing the condition to meet the requirements defined by the SLA, the QoS optimization rate of each candidate WS is computed based on performance, cost, and payload. Based on their optimization rate, the candidate WS are ordered and selected to accomplish the required task.
In [23] the authors propose both an evaluation approach for QoS attributes of WS, which is completely service and provider independent, and a method to analyze WS interactions and extract important QoS information without any knowledge about the service implementation.
In [24] the WS composition from a performance viewpoint is studied and measured. This measurements demonstrate that WS composition may reduce the maximal load of a system drastically (i.e., quasi-exponentially with the number of service compositions). In order to mitigate this performance reduction, the author proposes an optimized service composition architecture as a solution. This service component architecture uses service connectors on top of standard WS middleware. It optimizes automatically the local invocation of services with a Lookup-and-Service bus. The result is that no service congestion occurs since local service invocations have the cost of local call.
Proposals of QoS-aware frameworks can be found in [22, 25, 29]. Yu and Lin [29] present a broker-based framework for the dynamic integration of WS with end-to-end QoS constraints. The main functions of the proposed QoS broker include: service tracking, dynamic service composition model, dynamic service selection, and dynamic service adaptation. WebQ [22] is a QoS-based WS framework where the service selection is based on the parallel execution and monitoring of the candidate target services. Serhani et al. [25] propose a broker-based architecture which adopts QoS verification and certification in the service selection process.

5 Discussion

In Table 1 we summarize the different QoS approaches described in Section 4, according with the taxonomy defined in Section 3. The first column in the table identifies the approach. The "QoS Metric", "Point of View", "Dev. Level" and "QoS Enhancement Method" columns indicates the attributes already described in Section 3. In the "Dev. Level" column we use the abbreviations RT and DT to denote Run-Time and Design Time QoS management, respectively.

From our analysis we can derive some observations. Development, configuration and deployment of Web services can be a challenging task, which can require resources in term of manpower, time and hardware. The use of techniques which help delivering an adequate level of QoS is then crucial for the success of such systems. Design-time approaches allow both the development of QoS-aware services from the early design stages and the selection (composition) of existing services in such a way to satisfy performance, reliability, security and other extra-functional requirements. On the other hand, approaches which can be used at Run-time have the benefit of being able to cope with unexpected variabilities in the usage pattern (e.g., workload intensify), which are very difficult–almost impossible–to predict at design time. However, QoS management techniques which only work at Run-time can only be applied after all the services have been deployed, which can be difficult and costly.

The literature contains a large number of approaches dealing with QoS management at the consumer side. This is reasonable, as many QoS requirements are important mostly from the consumer point of view. However, customer-level requirements should be considered in conjunction with provider-level ones, which are quite different: while users are mostly interested, e.g., in reducing costs and decreasing response time, the service providers want to maximize their profits, resource usage and throughput (number of served requests per time unit). A successful QoS management strategy should consider both these point of views, and find an acceptable tradeoff between the customer requirements and the service provider ones. Regardless of their approach, techniques for the QoS management in WS should exhibit also the following basic characteristics:

Accuracy The prediction must be accurate enough in order to provide useful results. On the other hand, a compromise between the accuracy of predictions and the analysis effort must be found in order to enable the efficient evaluation of complex applications.
Adaptability Prediction and monitoring techniques should support efficient performance man-

agement under architecture changes where services are added/modified or replaced by different type of services.
Cost effectiveness The approaches should require less effort than prototyping and subsequent measurements and similarly the measurement of existing applications should take place in short time due to the high dynamism of the applications.
Scalability WS applications are typically built either with a large set of simple services or utilize few large-grain, complex services. To predict quality attributes, analysis techniques need to be scalable to handle both cases.
Generality The approach should be applicable to different component technologies with minimal modification. This enables the quality prediction of an integrated system with multiple component technologies involved.

6 Conclusions

In this paper we examined some of the current QoS analysis techniques for Web services mainly focusing on the performance attribute. We first defined a taxonomy of concerns which we used as a basis to classify the approaches described in the literature. Then, we applied this taxonomy to a set of QoS management techniques for WS, with special emphasis on architectural-level performance evaluation of WS applications. From the analysis of the existing approaches we derived some observations about strengths and weaknesses of different classes of approaches, and directions for future developments.

This work can be considered as a starting point towards a broader analysis which would include many other approaches which have been left outside from this paper. Anyway, we believe that the taxonomy we proposed here can be useful to classify the QoS analysis techniques for WS applications and to select a suitable analysis method.

References

[1] Qos for web services: Requirements and possible approaches. W3C Working Group Note, 25 November 2003, Available at `http://www.w3c.or.kr/kr-office/TR/2003/ws-qos/`.

[2] *International conference on service oriented computing*, 2003–2006.

[3] *IEEE international conference on web Services*, 2004–2006.

[4] D. Ardagna and B. Pernici. Global and local qos guarantee in web service selection. In C. Bussler and A. Haller, editors, *Business Process Management Workshops*, volume 3812, pages 32–46, 2005.

[5] G. Canfora, M. Di Penta, R. Esposito, and M. L. Villani. An approach for qos-aware service composition based on genetic algorithms. In *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*, pages 1069–1075, New York, NY, USA, 2005. ACM.

[6] V. Cardellini, E. Casalicchio, V. Grassi, and R. Mirandola. A framework for optimal service selection in broker-based architectures with multiple QoS classes. In *Services computing workshops, SCW 2006*, pages 105–112. IEEE computer society, 2006.

[7] J. Cardoso, A. P. Sheth, J. A. Miller, J. Arnold, and K. Kochut. Quality of service for workflows and web service processes. *J. Web Sem.*, 1(3):281–308, 2004.

[8] J. Cardoso, A. P. Sheth, J. A. Miller, J. Arnold, and K. J. Kochut. Modeling Quality of Service for Workflows and Web Service Processes. *Web Semantics J.: Science, Services and Agents on the World Wide Web*, 1(3):281–308, 2004.

[9] F. Casati, M. Castellanos, U. Dayal, and M.-C. Shan. Probabilistic, context-sensitive, and goal-oriented service selection. In *ICSOC '04: Proceedings of the 2nd international conference on Service oriented computing*, pages 316–321, New York, NY, USA, 2004. ACM.

[10] D. B. Claro, P. Albers, and J.-K. Hao. Selecting Web Services for Optimal Composition. In *Proc. of ICWS 2005 2nd Int'l Workshop on Semantic and Dynamic Web Processes*, Orlando, July 2005.

[11] C. H. Crawford and A. Dan. emodel: Addressing the need for a flexible modeling framework in autonomic computing. In *MASCOTS*, pages 203–208. IEEE Computer Society, 2002.

[12] W.-L. Dong and H. YU. Optimizing web service composition based on qos negotiation. In *Proc. EDOICS'06*, page 46, Los Alamitos, CA, USA, 2006. IEEE Computer Society.

[13] E. D. Lazowska, J. Zahorjan, G. S. Graham, and K. C. Sevcik. *Quantitative System Performance: Computer System Analysis Using Queueig Network Models*. Prentice-Hall, 1984.

[14] R. Levy, J. Nagarajarao, G. Pacifici, M. Spreitzer, A. N. Tantawi, and A. Youssef. Performance management for cluster based web services. In *Proc IM 2003*, pages 247–261. Kluwer Academic Publisher, 2003.

[15] G. Lodi, F. Panzieri, D. Rossi, and E. Turrini. Sla-driven clustering of qos-aware application servers. *IEEE Trans. Soft. Eng.*, 33(3):186–197, Mar. 2007.

[16] H. Ludwig. Web services qos: External slas and internal policies or: How do we deliver what we promise? In *Proceedings of the Fourth International Conference on Web Information Systems Engineering Workshops (WISE'03)*. IEEE, 2004.

[17] E. M. Maximilien and M. P. Singh. Conceptual model of web service reputation. *SIGMOD Rec.*, 31(4):36–41, 2002.

[18] E. M. Maximilien and M. P. Singh. A Framework and Ontology for Dynamic Web Services Selection. *IEEE Internet Computing*, 8(5):84–93, Sept./Oct. 2004.

[19] E. M. Maximilien and M. P. Singh. Toward Autonomic Web Services Trust and Selection. In *Proc. of 2nd Int'l Conf. on Service Oriented Computing*, pages 212–221, 2004.

[20] D. A. Menasce. QoS Issues in Web Services. *IEEE Internet Computing*, 6(6):72–75, Nov./Dec. 2002.

[21] D. A. Menasce. Composing Web Services: A QoS View. *IEEE Internet Computing*, 8(6):88–90, Nov./Dec. 2004.

[22] C. Patel, K. Supekar, and Y. Lee. A QoS Oriented Framework for Adaptive Management of Web Service Based Workflows. In *Proc. of DEXA 2003*, volume 2376 of *LCNS*, pages 826–835. Springer-Verlag, 2003.

[23] F. Rosenberg, C. Platzer, and S. Dustdar. Bootstrapping performance and dependability attributes ofweb services. In *ICWS '06: Proceedings of the IEEE International Conference on Web Services*, pages 205–212, Washington, DC, USA, 2006. IEEE Computer Society.

[24] H. A. Schmid. Service congestion: The problem, and an optimized service composition architecture as a solution. In *ICWS '06: Proceedings of the IEEE International Conference on Web Services*, pages 505–514, Washington, DC, USA, 2006. IEEE Computer Society.

[25] M. Serhani, R. Dssouli, A. Hafid, and H. Sahraoui. A QoS Broker Based Architecture for Efficient Web Services Selection. In *Proc. of 2005 Int'l Conf. on Web Services*, pages 113–120, Orlando, July 2005.

[26] G. Shercliff, J. Shao, W. A. Gray, and N. J. Fiddian. Qos assessment of providers with complex behaviours: An expectation-based approach with confidence. In *Proc. 4th Int. Conf on Service-Oriented Computing (ICSOC), Chicago, IL, USA, December 4-7*, volume 4294 of *LNCS*, pages 378–389. Springer, 2006.

[27] L.-H. Vu, M. Hauswirth, and K. Aberer. Qos-based service selection and ranking with trust and reputation management. In R. Meersman, Z. Tari, M.-S. Hacid, J. Mylopoulos, B. Pernici, Ö. Babaoğlu, H.-A. Jacobsen, J. P. Loyall, M. Kifer, and S. Spaccapietra, editors, *OTM Conferences (1)*, volume 3760 of *Lecture Notes in Computer Science*, pages 466–483. Springer, 2005.

[28] T. Yu and K.-J. Lin. The design of qos broker algorithms for qos-capable web services. In *EEE '04: Proceedings of the 2004 IEEE International Conference on e-Technology, e-Commerce and e-Service (EEE'04)*, pages 17–24, Washington, DC, USA, 2004. IEEE Computer Society.

[29] T. Yu and K.-J. Lin. A broker-based framework for qos-aware web service composition. In *EEE '05: Proceedings of the 2005 IEEE International Conference on e-Technology, e-Commerce and e-Service (EEE'05) on e-Technology, e-Commerce and e-Service*, pages 22–29, Washington, DC, USA, 2005. IEEE Computer Society.

[30] T. Yu and K. J. Lin. Service Selection Algorithms for Composing Complex Services with Multiple QoS Constraints. In *Proc. of 3rd Int'l Conf. on Service Oriented Computing*, pages 130–143, Amsterdam, The Netherlands, Dec. 2005.

[31] L. Zeng, B. Benatallah, A. H. H. Ngu, M. Dumas, J. Kalagnanam, and H. Chang. QoS-Aware Middleware for Web Services Composition. *IEEE Trans. Softw. Eng.*, 30(5):311–327, Aug. 2004.

Approach	QoS Metric	Point of View	Dev. Level	QoS Enhancement Method
Cardoso et al. [7]	Resp. Time, Cost, Reliability	Consumer	DT	Models
Maximilien et al. [18, 19]	Reputation, Trust	Consumer	DT, RT	Ontology
Vu et al. [27]	Resp. Time, Trust, Reputation	Consumer	DT, RT	Ontology
Casati et al. [9]	Resp. Time	Consumer	RT	Data mining on measured data
Zeng et al. [31]	Price, Availability, Reliability, Reputation	Consumer	DT	Optimization models
Ardagna et al. [4]	Resp. Time, Cost, Reputation	Consumer	DT, RT	Optimization models
Cardellini et al. [6]	Resp. Time, Cost, Reputation	Consumer	DT	Optimization models
Canfora et al. [5]	Resp. Time, Cost, Availability, Reliability	Consumer	DT	Models (genetic algorithms)
Claro et al. [10]	Resp. Time, Cost, Availability, Reliability	Consumer	DT	Models (genetic algorithms)
Yu and Lin. [30]	Multiple QoS	Consumer	DT	Optimization models
Yu and Lin. [29]	Resp. Time	Consumer	RT	Broker-based framework and measurement
Patel [22]	Multiple QoS	Consumer	RT	Service Discovery, Selection and Monitoring
Serhani et al. [25]	Resp. Time, Latency, Availability, Reputation, Price	Consumer	RT	Broker-based framework and measurement
Dong and Yu [12]	Resp. Time, Reliability, Cost	Consumer	DT	Optimization models
Rosenberg et al. [23]	Latency, Resp. Time, Availability, Accuracy	Consumer	RT	Measurement
Schmid [24]	Latency	Consumer	RT	Measurement
Levy et al. [14]	Resource Utilization, Resp. Time	Consumer and Provider	DT and RT	Dispatching and Monitoring
Crawford et al. [11]	Resource Utilization, Resp. Time	Consumer and Provider	DT and RT	Dispatching and Monitoring
Shercliff et al. [26]	Multiple QoS	Provider	DT and RT	Monitoring and models
Yu and Lin [28]	Multiple QoS	Consumer and Provider	RT	Broker-based
Panzieri et al. [15]	Resp. Time, Availability, Efficiency[3]	Consumer and Provider	RT	Dynamic Service Reconfiguration and Monitoring

Table 1. Classification of the approaches