2014 IEEE 10th World Congress on Services

Aggregate QoS Estimation of Service Compositions – An Analysis of Pattern-oriented Approaches

Perla Velasco-Elizondo
Autonomous University of Zacatecas, UAZ.
Zacatecas, ZAC, 98000, Mexico.
pvelasco@uaz.edu.mx

David Barredo-Hernandez, Hugo A. Mitre
Centre for Mathematical Research, CIMAT.
Zacatecas, Zac., 98060, Mexico
{davofredo, hmitre}@cimat.mx

***Abstract*—For service-oriented systems the estimation of QoS is an important factor to determine whether a prospective service composition will meet the expectations of its final users. Many approaches to QoS estimation of service compositions have been proposed. However, not all of them allow the QoS of composition be estimated a priori and systematically from the QoS of its individual services. This is an important goal for many composition-based development approaches. This paper is a first effort to analyze QoS estimation approaches aligned to this goal. We revise approaches that use patterns as composition mechanisms as they enable the QoS of a service composition be estimated, a priori and systematically, by aggregating the QoS information of its constituent services. The result of this analysis offers insight in the main patterns and quality attributes addressed by these approaches as well as the adopted aggregation criteria. This will help us to achieve our ultimate goals of using these approaches on public services, assessing them for practicality, identifying gaps between theory and practice and possibilities for future work.**

***Keywords*-service composition; quality of service estimation; service composition patterns.**

I. Introduction

Building software systems by composing pre-exiting software components has become a popular approach to software development.[1] Nowadays there exist a large number of repositories of reusable components and supporting development environments to compose them, e.g., [33], [23], [32]. In this context, Service-Oriented Architecture (SOA) [9] has emerged as a development paradigm to build software systems by composing components that take the form of *Web services*.[2] In recent years the adoption of this paradigm has grown greatly to the point that *Web service composition*, or service composition for short, is today a main topic in academia and industry [19].

For software systems the term *quality of service* (QoS) has been traditionally used to denote the values of various *quality attributes*, e.g., scalability, availability. The estimation of QoS is an important aspect for service composition to determine whether a prospective composition will meet the expectations of its final users. Many approaches to estimation of QoS of service-oriented systems have been proposed. However, we are particularly interested on those in which the QoS of a service composition is estimated *a priori* and *systematically* from the QoS of its individual services. The reason of the former is because we consider that these approaches are more aligned to *predictable assembly* [11], which is an important goal for many composition-based development approaches.

This paper is a first effort to analyze approaches to QoS estimation aligned to this goal. We focus on approaches that use *patterns* as composition mechanisms. Patterns are predefined compositional structures used to support systematic system design. Regarding service composition, they enable the QoS of a service composition be estimated, a priori and systematically, by aggregating the QoS information of its constituent services. Moreover, the use patterns for composing reusable services is very popular in real practice; it is well-know that many patterns can be implemented using popular service composition languages, see [37], [38], [24]. In this context, we are interested in identifying what has been most addressed in past research. This will help us to achieve our ultimate goals of using these approaches on public services, assessing them for practicality, identifying gaps between theory and practice and possibilities for future work. Because of the former, in this analysis we focus on providing answers to the following questions:

(1) what are the most addressed composition patterns?
(2) what are the most addressed quality attributes?
(3) what types of aggregation criteria are used?

This paper is organized as follows. In Section II we give an overview of the background related information, including related work. In Section III we describe the aspects we take into account to analyze the QoS estimation approaches; we call them analysis dimensions. In Section IV we present the results of the analysis considering the previously described dimensions. Finally, in Section V we present the conclusions and future work.

[1]In this paper we use the term software component in a generic manner to refer to a unit of computation that can be subject of composition via specific composition mechanisms.

[2]We assume familiarity with Web services and related implementation technologies.

978-1-4799-5069-0/14 $31.00 © 2014 IEEE
DOI 10.1109/SERVICES.2014.70

Table I: Quality attributes included in some quality models for Web services proposals.

	Performance						**Trustworthiness**								**Security**										**Economic**		**Usability**					**Other**						
Reference	Response Time	Processing Time	Latency	Throughput	Scalability, Capacity	Resource Utilization	Availability	Accessibility	Accuracy	Reliability, Successability	Error handling	Robustness, Flexibility	Guaranteed messaging	Reputation	Authentication	Authorization	Integrity	Confidentiality, Privacy	Accountability	Traceability, Auditability	Non-repudiation	Encryption	Observability	Safety	Cost, Price	Penalty and Incentive	Efficiency of Use	Content Accessibility	Learnability, Understandability	Satisfaction	Effectiveness	Standard Conformability	Maintainability	Reusability	Stability	Completeness	Interoperability	Recognition, Discoverability
Ran [27]	x		x	x	x		x		x	x	x	x	x		x	x	x	x	x	x	x	x			x							x			x	x		
Cappiello et al. [5]	x	x	x	x	x		x	x	x	x	x	x		x	x	x	x	x	x	x	x	x		x	x		x	x	x	x	x	x			x	x		
Muñoz et al. [25]	x		x	x	x	x	x	x	x	x			x			x		x	x	x	x	x			x										x		x	
Balfagih and Hassan [2]	x			x			x			x		x		x	x		x	x	x										x			x	x	x			x	x
OASIS [26]	x	x		x	x		x	x		x			x	x	x	x	x	x		x	x	x	x	x	x	x						x				x	x	x

II. Background Information

A. Quality Attributes

When developing software systems satisfying functional requirements is not enough. Systems users also have requirements about how well those systems should work. These requirements are known as quality attributes.

Quality attributes are difficult to define and categorize. For traditional systems there exist several quality models providing each one its own definitions and classifications for quality attributes, e.g., ISO/IEC 9126 [15] or Dromey's quality model [7]. For service-oriented systems some quality model proposals have started to emerge. Table I shows a list of quality attributes considered in some of these proposals. For convenience we have grouped the quality attributes into categories namely, performance, trustworthiness, security, economic, usability and other.

B. Service Composition Patterns

In any view of composition, composition mechanisms compose components into larger pieces of software [21]. For Web services a common approach to define a composition is to model it as a behavior that results from coordinating the execution of a set of them [36], [21]. When doing so, this resulting behavior is often described by a pattern. As introduced before, patterns denote pre-defined compositional structures to support systematic system design. There is a large number of patterns, e.g., [10], [30], [34]; the patterns to which we refer in this paper denote compositional structures which define the order in which a collection of services' operations are executed. That is, the control-flow. For simplicity, in this paper we call them *composition patterns*.

Figure 1 a) shows a graphical representation of one of these patterns, which is known as the AND Pattern. This pattern denotes a control flow structure where a single flow of control diverges at some point to allow the execution of the operations of two or more services in parallel and eventually converges after all executions have been completed. Table II shows descriptions of some other well-known patterns – including the AND Pattern, that are often used to support service composition. Note that these patterns are used to model the structure of a whole composition and that many of them are inspired by the workflow patterns defined in [30], [35].

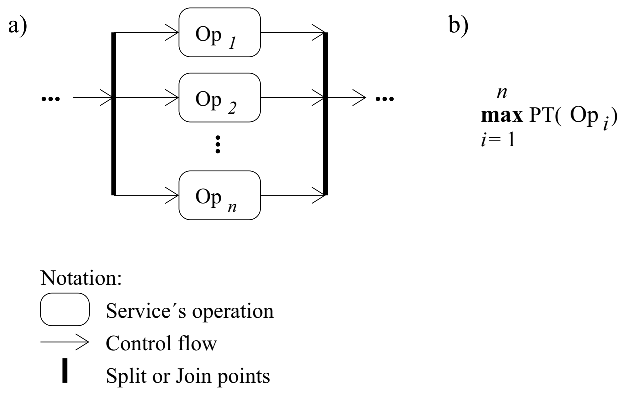

Figure 1: a) Graphical representation of the AND Pattern and b) an example of an aggregation function.

As introduced before, the use patterns as composition operators for services is very popular nowadays as many patterns can be implemented using popular service composition languages with few effort.

C. QoS Aggregation

When using patterns to compose services aggregation is a common method to estimate, a priori and systematically,

Table II: Patterns that are often used for service composition.

Name	Description
Sequence	It denotes a control flow structure where the operations of two or more services are executed in a sequential order.
AND	It denotes a control flow structure where a single flow of control diverges at some point to allow the execution of the operations of two or more services in parallel. The flow eventually converges after all executions have completed.
OR	It denotes a control flow structure where a single flow of control diverges at some point to allow the execution of the operations of two or more services based on the evaluation of a boolean expression. The flow eventually converges after all executions have completed.
XOR	It denotes a control flow structure where a single flow of control diverges at some point to allow the execution of only one service's operation, from a set of two or more, based on the evaluation of a boolean expression. The flow eventually converges after the service's operation has completed.

the QoS of a composition from its constituent services. QoS aggregation is performed by using *aggregation functions* that are generally defined based on the semantics of both, the addressed quality attribute and the pattern used to define the composition.

Figure 1 b) shows an example of aggregation function to estimate the *processing time* quality attribute. In this example, it is assumed that the composition is defined in terms of the AND Pattern –depicted in Figure 1 a). Thus, taking into consideration the semantics of both, the quality attribute and the pattern, the composition's processing time is the maximum value of the processing times PTs of the operations Op in the n parallel branches.

D. Related Work

There exists various works tackling QoS aspects of individual services and their compositions. Surprisingly we found only three relevant surveys considering QoS estimation approaches that use patterns for service composition [20], [19], [31]. However, as the main focus of these surveys is not on patterns, from the described results it is not possible to know what are the most addressed patterns and quality attributes by these approaches. Similarly, few consideration is given to identify the types of aggregation criteria are used by them.

We have mentioned that the use patterns for composing reusable services is popular in real practice. Although the information provided by these surveys contribute to gain a better understanding of the aspects related to the problem of QoS estimation of service compositions, it is limited for a practitioner wanting, for example, to know if there is a commonly used aggregation criteria for a quality attribute when performing pattern-based composition of Web services.

III. Analysis Dimensions and Reviewed Works

As introduced before, we are interested in identifying the main patterns and quality attributes addressed in past research as well as the adopted aggregation criteria. Thus, we defined some analysis dimensions that are both easy to manage and relevant to our interest. In this section we describe them.

A. Analysis Dimensions

Composition Patterns Types. Composition patterns can be classified considering a variety of characteristics. Regarding service composition, we classify patterns as follows:

a) Basic Patterns, which define compositional structures that capture basic styles of control-flow. In this category we include the patterns listed in Table II.

b) Advanced Patterns, which define compositional structures that capture more advanced styles of control-flow. Any other pattern that is not included in the former category is included here, i.e. Discriminator, Interleaved Parallel Routing, Request/Reply, Synchronous Polling [37].

Quality Attributes Types. A great number of quality attributes exist and they can be classified in many different manners. For the purposes of this work will consider the categories and attributes shown in Table I:

a) Performance related attributes, which relate to the responsiveness of a composition to execute a computation within a given time interval.

b) Trustworthiness related attributes, which relate to the confidence that a composition will behave as expected in normal use.

c) Security related attributes, which relate to the ability of a composition to protect itself from intrusion.

d) Economic related attributes, which relate to the amount of money paid for using a composition.

e) Usability related attributes, which relate to how well users can use a composition.

f) Other attributes, which are those that are not included in the former categories.

Aggregation Criteria. Aggregation functions, which are the means to support aggregate QoS estimation, are defined using some criteria. In this work we will consider the following:

a) Basic aggregation, which results in an aggregation function that takes several values of a same quality attribute. The aggregation function for estimating the processing time shown in Figure 1 b) illustrates this criterion. As can be seen, the function **max** takes processing time values of each constituent service's operation involved.

b) Derived aggregation, which results in an aggregation function that takes several values of different quality attributes. For example, an aggregation function to estimate the response time of a service composition could take values of attributes such as latency and processing time of each service's operation involved.

d) Usage-based aggregation, which results in an aggregation function that takes values of not only various and maybe different quality attributes, but also values related to the particular use of a composition. For instance, by taking the execution probability value of each service's operation involved in the composition, the function for estimating the processing time shown in Figure 1 b) could become usage-based.

B. Reviewed Works

The analyses presented in this paper were performed on QoS estimation proposals published in 19 works. These works are shown in Table III.

Table III: List of reviewed works.

Reference	**Year**	**Citations**
Zeng et al. [40]	2003	1000+
Zeng et al. [41]	2004	2000+
Cardoso [6]	2004	800+
Jaeger et al. [16]	2004	400+
Jaeger et al. [17]	2005	400+
Berbner et al. [3]	2006	200+
Hwang et al. [14]	2007	100+
Ko et al. [18]	2008	100+
Ma and Chen [22]	2008	10+
Rosario et al.[28]	2008	90+
Alrifai and Risee [1]	2009	300+
Rosenberg et al. [29]	2009	50+
Hu et al. [13]	2009	2
Dumas et al. [8]	2010	30+
Zheng, et al. [42]	2010	20+
Hu et al. [12]	2010	8
Bhuvaneswari [4]	2011	1
Xu et al. [39]	2012	1
Zheng et al. [43]	2013	7

For selection and analysis of the revised works we followed the process depicted in Fig. 2. After defining our scoping questions –introduced in Section I, candidate works were identified by using search strings (e.g. "QoS" AND "service composition" AND "patterns") on relevant databases for computer science and software engineering (e.g. IEEE Xplore and ACM Digital Libraries). Next, for deciding inclusion of a work, we considered papers published within a 10-year time frame from 2003 to 2013 using pre-defined compositional structures as means to service composition. Being this a first effort, we decided to include a maximum of 3 papers by year preferring those with (more) citations. Driven by the analysis dimensions defined in Section III-A, next we performed data extraction and analysis from the included works. Finally, we visualized the obtained results which we describe next.

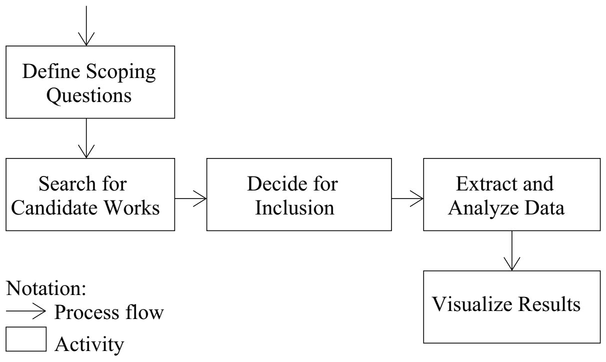

Figure 2: Adopted process for selection and analysis of the revised works.

IV. Analysis Results

A. Composition Patterns Types

Figure 3 shows a graph that lists the most popular composition patterns addressed in the reviewed works. As can be seen, except for the Discriminator Pattern, most of the addressed patterns belong to the Basic Patterns category –described in Section III-A. The patterns more addressed are the Sequence and AND Patterns –in 17 and 14 works respectively. After these patterns comes the XOR Pattern –which is addressed in 9 works. Finally, the less popular ones are the Discriminator and OR Patterns –both addressed in 4 works.

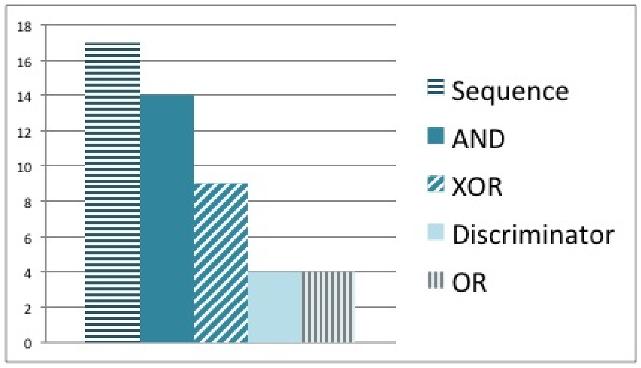

Figure 3: Popular composition patterns addressed in the reviewed works.

Web service composition languages provide constructs that allow the direct implementation of various composition patterns [38], [24]. For example constructs such as `<sequence>` and `<all>`, or some other syntactic variants, are available in many composition languages and allow for sequential and parallel execution of services' operations respectively. Thus, creating compositions using the Sequence and AND Patterns is common and easy. This could be the reason why these patterns are the most considered in QoS estimation proposals.

The XOR and OR Patterns can be also implemented using the constructs of popular service composition languages. However, estimating the QoS of service compositions using these patterns often requires knowing the execution probability of each service's operation in the participating branches. The fact that in practice this information is not always known could explain the reduction of the number of works considering these patterns. Additionally, and in contrast to the XOR Pattern, the OR Pattern cannot always be directly implemented using composition languages constructs, see [38], [24]. This could explain why less works consider this pattern compared to the ones that do for the XOR Pattern.

Finally, the Discriminator Pattern is an advanced pattern. It denotes a compositional structure where a set of two or more service's operations converge into a subsequent single control-flow once the first incoming service's operation has been completed. It has been also discussed that this pattern cannot be directly implemented using common composition languages constructs, see [38], [24]. Thus, this could explain why the number of works considering this pattern is also low.

B. Quality Attributes Types

In section II-A we mentioned that proposals of quality models for Web services have started to emerge. We also showed the quality attributes included in some of them. However, we discover that the revised QoS estimation approaches do not use them at all. In fact, all these works only focus on defining aggregation functions for a set of well-known quality attributes.

Figure 4 depicts the top four quality attributes categories considered in the reviewed works namely *performance* –in 18 works, *trustworthiness* –in 15 works, *economic* –in 13 works, and *security* –in 2 works. Additionally, for analysis purposes, in Table IV we have highlighted the specific quality attributes in these categories for which aggregation functions were found. We use bolds to denote this.

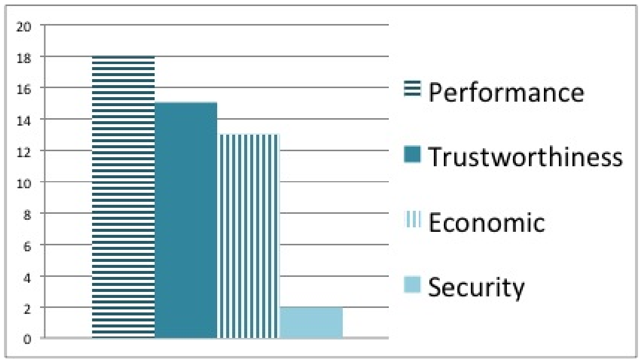

Figure 4: Popular quality attributes addressed in the reviewed works.

Regarding performance related attributes, in the reviewed works it was very common to find aggregation functions for estimating *latency*, *response time* and *throughput*. As Table IV shows for the performance category not all the specific attributes proposed in Web service quality models are addressed. However, for the two attributes present in all quality models revised, i.e. response time and latency, aggregation functions were found.

As Table IV shows, few attributes in the trustworthiness category are addressed. In the reviewed works aggregation functions to estimate *reliability*, *availability* and *reputation* were found. In these works reliability is defined as the probability of a service composition to deliver computations as expected; availability was defined as the probability of a service composition to deliver computations when requested; and reputation was defined as the average perception of the service composition by end users. Note that, as in previous case, for the two attributes that are present in all quality models revised, i.e. reliability and availability, aggregation functions were also found.

In the reviewed works aggregation functions for cost and price were proposed to estimate economic related quality attributes.

Regarding security related attributes, the work that addressed this category did it by estimating *encryption level*. Despite the fact that this category contains several quality attributes, as Table IV shows, only this attribute was addressed by the revised estimation approaches.

As Web services are black boxes capable of performing specific operations, there is a tendency to estimate quality attributes that are relevant for final users of service compositions. However, it is strange that none of the revised works provided aggregation functions to estimate attributes in the usability category. Finally, the fact that attributes such as maintainability, reusability, interoperability, and some others of this kind, are not addressed may denote that the resulting compositions are not composite Web services. Composite Web services are meant to be (re)used, as the atomic ones, for the construction of service-based systems.

C. Aggregation Criteria

By analizing all the revised works we ended up with a set of 73 aggregation functions. Figure 5 shows the kind of aggregation criteria to which these functions belong. As can be observed, the functions are distributed as follows: 72 are basic aggregation functions, 20 are usage-based aggregation functions and 0 are derived aggregation functions. Note that the usage-based aggregation functions found are also counted in the basic aggregation set. As explained in Section III-A, a function such as

$$\sum_{i=1}^{n} p_i \cdot Lat(Op_i),$$

Table IV: Quality attributes in Web service quality model proposals for which aggregation functions were found.

	Performance						Trustworthiness								Security										Economic		Usability					Other						
Reference	**Response Time**	Processing Time	**Latency**	**Throughput**	Scalability, Capacity	Resource Utilization	**Availability**	Accessibility	Accuracy	**Reliability, Successability**	Error handling	Robustness, Flexibility	Guaranteed messaging	**Reputation**	Authentication	Authorization	Integrity	Confidentiality, Privacy	Accountability	Traceability, Auditability	Non-repudiation	**Encryption**	Observability	Safety	**Cost, Price**	Penalty and Incentive	Efficiency of Use	Content Accessibility	Learnability, Understandability	Satisfaction	Effectiveness	Standard Conformability	Maintainability	Reusability	Stability	Completeness	Interoperability	Recognition, Discoverability
Ran [27]	x		x	x	x		x		x	x	x	x	x		x	x	x	x	x	x	x	x			x							x			x	x		
Cappiello et al. [5]	x	x	x	x	x		x	x	x	x	x	x		x	x	x	x	x	x	x	x	x		x	x		x	x	x	x	x	x			x	x		
Muñoz et al. [25]	x		x	x	x	x	x	x	x	x			x			x		x	x	x	x	x			x										x		x	
Balfagih and Hassan [2]	x			x			x			x		x		x	x		x	x	x										x			x	x	x			x	x
OASIS [26]	x	x		x	x		x	x		x			x	x	x	x	x	x		x	x	x	x	x	x	x						x				x	x	x

to estimate the latency of a composition as the summation of the latencies Lat of the n involved operations Op times the execution probability p of such operation, is considered as both basic and usage-based. The function is basic because it takes values of several quality attributes of the same type –i.e. the latencies Lat; it also takes a value related to the particular use of the composition –i.e. the execution probability p of each operation.

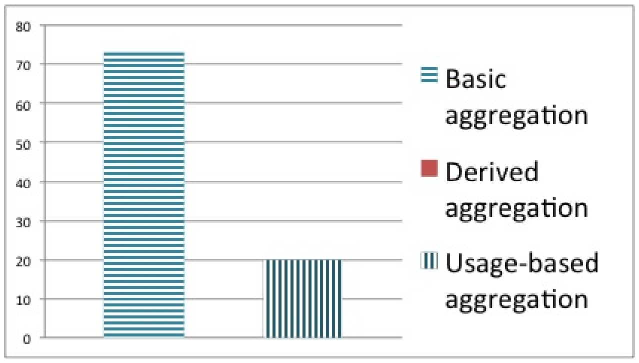

Figure 5: Aggregation criteria in the set of aggregation functions.

Regarding basic aggregation, there have been defined aggregation functions for all colour highlighted attributes in Table IV, i.e., *response time*, *latency*, *throughput*, *availability*, *reliability*, *reputation*, *encryption* and *cost*. Although all patterns in Figure 3 are considered, most of the functions adopting this aggregation criterion were defined for the Sequence and AND Patterns.

Except for *encryption*, usage-based aggregation functions for all colour highlighted attributes in Table IV have been also defined. In this case, most of these functions were defined for the XOR Pattern. The value related to the particular use of the composition was always the execution probability of each service's operation involved in the composition.

It is very surprising that none of the proposals considered a derived aggregation criteria for the definition of aggregation functions. We consider that the lack of a standard Web service quality model as well as the heterogeneity of current proposals complicate the understating of the relationships among quality attributes. A clear semantics about the quality attributes and their relationships is an important aspect to define derived aggregation functions.

Finally, another aspect noted in this analysis is that four types of aggregation operations were mainly used in this functions: summation, multiplication, the minimum relation and the maximum relation. This finding, together with that of existing mostly basic aggregation functions could be insufficient for the precise evaluation of the QoS of a service composition.

D. *Other Aspects*

Another interesting finding is that in most of the revised works it is assumed a composition scenario where developers are concerned about one quality attribute only. We believe that this scenario is limited, as software systems in practice require supporting multiple quality attributes. Thus, for developers it is a realistic requirement to have access to means to estimate the QoS of a service composition considering multiple QoS concerns.

V. Summary and Future Work

In this paper we have presented the results of an analysis of QoS estimation approaches that use patterns as composition mechanisms. These approaches allow an a priori and systematic estimation of the QoS of a service composition by aggregating the QoS information of its constituent services by using aggregation functions.

The results of the analysis offered insight in the main patterns and quality attributes addressed by these approaches as well as the adopted aggregation criteria. Regarding patterns, the Sequence and AND Patterns are the more recurrent in the reviewed works. Regarding quality attributes, it was very common to find aggregation functions for estimating latency, response time and throughput. The basic aggregation criterion was the one adopted by the majority of the works. This aggregation criterion results in an aggregation functions that take values of several quality attributes of the same type. Four types of aggregation operations were mainly used in this functions: summation, multiplication, the minimum relation and the maximum relation. We did not find approaches defining derived aggregation functions, which allow for estimating quality attributes that depend on values of other different attributes, e.g. response time could depend on other quality values such as latency and processing time. Another interesting finding is that most of the revised works assume a composition scenario where developers are concerned about one quality attribute only. In practice, however, software systems require supporting multiple quality attributes.

Our future work includes reviewing other approaches to QoS estimation of service compositions in order to validate our current findings. We also plan to perform some other analyses. For example, determining the relationship between (i) patterns, quality attributes and aggregation criteria and (ii) the nature of the optimization method when multiple quality attributes are considered. We also plan to consider to identify more aggregation criteria. For instance, it seems to be necessary to have an aggregation criteria that considers other architectural related aspects beyond composition patterns, e.g., values that apply to the channels that support service communication. Additionally, and based on the obtained results in this works, we are currently working on assessing the applicability of the aggregation functions on services in public repositories as many of the works reviewed applied the functions to locally generated examples.

References

[1] M. Alrifai and T. Risse. Combining Global Optimization with Local Selection for Efficient QoS-aware Service Composition. In *Proceedings of the 18th International Conference on World Wide Web*, pages 881–890, New York, NY, USA, 2009. ACM.

[2] Z. Balfagih and M.F. Hassan. Quality model for web services from multi-stakeholders' perspective. In *Proceedings of the 2009 International Conference on Information Management and Engineering*, pages 287–291, Washington, DC, USA, 2009. IEEE Computer Society.

[3] R. Berbner, M. Spahn, N. Repp, O. Heckmann, and R. Steinmetz. Heuristics for QoS-aware Web Service Composition. In *Proceedings of the IEEE International Conference on Web Services*, ICWS '06, pages 72–82, Washington, DC, USA, 2006. IEEE Computer Society.

[4] A. Bhuvaneswari. QoS Considerations for a Semantic Web Service Composition. *European Journal of Scientific Research*, 65(3):403–415, 2011.

[5] C. Cappiello, K. Kritikos, A. Metzger, M. Parkin, B. Pernici, P. Plebani, and M. Treiber. A quality model for service monitoring and adaptation. In *In Proceedings of the Workshop on Monitoring, Adaptation and Beyond*, pages 29–42, 2008.

[6] J. Cardoso, J. Miller, A. Sheth, and J. Arnold. Quality of service for workflows and web service processes. *Journal of Web Semantics*, 1:281–308, 2004.

[7] R.G. Dromey. A model for software product quality. *Transactions on Software Engineering*, 21(2):146–162, 1995.

[8] M. Dumas, L. Garca-Bauelos, A. Polyvyanyy, Y. Yang, and L. Zhang. Aggregate quality of service computation for composite services. In P. Maglio, M. Weske, J. Yang, and M. Fantinato, editors, *Service-Oriented Computing*, volume 6470 of *Lecture Notes in Computer Science*, pages 213–227. Springer Berlin Heidelberg, 2010.

[9] T. Erl. *Service-Oriented Architecture: Concepts, Technology, and Design*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2005.

[10] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Professional Computing Series, 1995.

[11] D. Hamlet. Defining predictable assembly. In I. Gorton, G. Heineman, I. Crnkovic, H. Schmidt, J. Stafford, C. Szyperski, and K. Wallnau, editors, *Component-Based Software Engineering*, volume 4063 of *Lecture Notes in Computer Science*, pages 320–327. Springer Berlin / Heidelberg, 2006.

[12] T. Hu, M. Guo, S. Guo, H. Ozaki, L. Zheng, K. Ota, and M. Dong. MTTF of Composite Web Services. In *Proceedings of the International Symposium on Parallel and Distributed Processing with Applications*, pages 130–137, Los Alamitos, CA, USA, 2010. IEEE Computer Society.

[13] T. Hu, S. Guo, M. Guo, F. Tang, and M. Dong. Analysis of the availability of composite web services. In *Proceedings of the International Conference on Frontier of Computer Science and Technology*, pages 231–237, Washington, DC, USA, 2009. IEEE Computer Society.

[14] S-Y. Hwang, H. Wang, J. Tang, and J. Srivastava. A probabilistic approach to modeling and estimating the QoS of web-services-based workflows. *Information Sciences*, 177(23):5484–5503, 2007.

[15] *International Standard 9126. Information Technology-Software Product Evaluation – Quality characteristics and guidelines for their use*. International Organization for Standardization and International Electrotechnical Commission, 1991.

[16] M.C. Jaeger, G. Rojec-goldmann, and G. Muehl. QoS Aggregation for Web Service Composition using Workflow Patterns. In *Proceedings of the IEEE International Conference on Enterprise Distributed Object Computing*, pages 149–159. IEEE Computer Society, 2004.

[17] M.C. Jaeger, G. Rojec-goldmann, and G. Muehl. QoS Aggregation in Web Service Compositions. In *Proceedings of the IEEE International Conference on e-Technology, e-Commerce, and e-Services*, pages 181–185, Los Alamitos, CA, USA, 2005. IEEE Computer Society.

[18] J.M. Ko, C.O. Kim, and I.-H. Kwon. Quality-of-service oriented web service composition algorithm and planning architecture. *Journal of Systems and Software*, 81(11):2079 – 2090, 2008.

[19] O. Kondratyeva, N. Kushik, A.R. Cavalli, and N. Yevtushenko. Evaluating quality of web services: A short survey. In *Proceedings of the IEEE 20th International Conference on Web Services*, pages 587–594. IEEE Computer Society, 2013.

[20] K. Kritikos, B. Pernici, P. Plebani, C. Cappiello, M. Comuzzi, S. Benrernou, I. Brandic, A. Kertész, M. Parkin, and M. Carro. A survey on service quality description. *ACM Computing Surveys*, 46(1):1:1–1:58, July 2013.

[21] K.-K. Lau and T. Rana. A taxonomy of software composition mechanisms. In *Proc. 36th EUROMICRO Conference on Software Engineering and Advanced Applications*, pages 102–110. IEEE, 2010.

[22] J. Ma and H. Chen. A reliability evaluation framework on composite web service. In *Proceedings of the IEEE International Symposium on Service-Oriented System Engineering*, pages 123 –128. IEEE Computer Society, 2008.

[23] Membrane. Directory of Public SOAP Web Services. http://www.servicerepository.com.

[24] S. Migliorini, M. Gambini, M. La Rosa, and A.H.M. ter Hofstede. Pattern-based evaluation of scientific workflow management systems, February 2011. http://eprints.qut.edu.au/39935/.

[25] H. Munoz, I. Kotsiopoulos, L.M. Gonzalez, and L. Merino. Enhancing Service Selection by Semantic QoS. In *6th Annual European Semantic Web Conference (ESWC2009)*, pages 565–577, June 2009.

[26] OASIS. Web Services Quality Factors Version 1.0. Candidate OASIS Standard 01, October 2012. http://docs.oasis-open.org/wsqm/WS-Quality-Factors/v1.0/WS-Quality-Factors-v1.0.pdf.

[27] S. Ran. A Model for Web Services Discovery with QoS. *SIGecom Exch.*, 4(1):1–10, March 2003.

[28] S. Rosario, A. Benveniste, S. Haar, and C. Jard. Probabilistic QoS and Soft Contracts for Transaction-Based Web Services Orchestrations. *IEEE Transactions on Services Computing*, 1(4):187–200, 2008.

[29] F. Rosenberg, P. Celikovic, A. Michlmayr, P. Leitner, and S. Dustdar. An end-to-end approach for QoS-aware service composition. In *Proceedings of the IEEE International Conference on Enterprise Distributed Object Computing*, pages 128–137. IEEE Press, 2009.

[30] N. Russell, A.H.M. ter Hofstede, W.M.P. van der Aalst, and N. Mulyar. Workflow control-flow patterns: A revised view. Technical Report BPM-06-22, BPM Center, 2006.

[31] A. Strunk. Qos-aware service composition: A survey. In *Proceedings of the IEEE European Conference on Web Services*, pages 67–74. IEEE Computer Society, 2010.

[32] The Laboratory of Neuro Imaging at the University of California at Los Angeles. LONI Pipeline. http://pipeline.bmap.ucla.edu/.

[33] The University of Manchester and the European Bioinformatics Institute (EMBL-EBI). BioCatalogue. The Life Science Web Services Registry. http://www.biocatalogue.org/.

[34] E. Thomas. *SOA Design Patterns*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1st edition, 2009.

[35] W.M.P. van der Aalst, A.H.M. ter Hofstede, B. Kiepuszewski, and A.P. Barros. Workflow patterns. In *Distributed and Parallel Databases*, pages 5–51, 2003.

[36] A. W.M.P. van der, M. Dumas, and H. A.H.M. ter. Web Service Composition Languages: Old Wine in New Bottles? In *Proceeding of the 29th EUROMICRO Conference: New Waves in System Architecture*, pages 298–305. IEEE Computer Society, Los Alamitos, CA, 2003.

[37] P. Wohed, W.M.P. van der Aalst, M. Dumas, and A.H.M. ter Hofstede. Analysis of Web Services Composition Languages: The Case of BPEL4WS. In *Proceedings of the 22nd International Conference on Conceptual Modeling*, volume 2813 of *Lecture Notes in Computer Science*, pages 200–215. Springer Verlag, 2003.

[38] P. Wohed, W.M.P van der Aalst, M. Dumas, A.H.M. ter Hofstede, and N. Russell. Pattern-based analysis of BPMN – an extensive evaluation of the control-flow, the data and the resource perspectives (revised version). Technical Report BPM-06-17, BPMcenter.org, 2006.

[39] J. Xu, H. Sun, X. Wang, X. Liu, and R. Zhang. Optimizing pipe-like mashup execution for improving resource utilization. In *Proceedings of the Fifth IEEE International Conference on Service-Oriented Computing and Applications*, pages 1–4, Los Alamitos, CA, USA, 2012. IEEE Computer Society.

[40] L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam, and Q. Sheng. Quality driven web services composition. In *Proceedings of the 12th International Conference on World Wide Web*, pages 411–421, New York, NY, USA, 2003. ACM.

[41] L Zeng, B. Benatallah, A. H.H. Ngu, M. Dumas, J. Kalagnanam, and H. Chang. QoS-Aware Middleware for Web Services Composition. *IEEE Transactions of Software Engineering*, 30(5):311–327, May 2004.

[42] H. Zheng, J. Yang, and W. Zhao. QoS Analysis and Service Selection for Composite Services. pages 122–129. IEEE Computer Society, 2010.

[43] H. Zheng, W. Zhao, J. Yang, and A. Bouguettaya. QoS Analysis for Web Service Compositions with Complex Structures. *IEEE Trans. Serv. Comput.*, 6(3):373–386, 2013.