A Survey on Approaches for Timed Services

Kristian Duske[1] and Richard Müller[2]

[1] Institut für Softwaretechnik und Theoretische Informatik
Technische Universität Berlin, Germany
kristian.duske@tu-berlin.de
[2] Institut für Informatik, Humboldt-Universität zu Berlin, Germany
richard.mueller@informatik.hu-berlin.de

Abstract. In the context of service-oriented computing, time has been extensively studied in literature. We present a survey on possible problem statements for timed services, and give an overview of state-of-the-art approaches. Thereby we identify which problems are already thoroughly researched and which problems warrant further research.

Keywords: SOC, timed services, behavior, quality of service, survey

1 Introduction

Service-oriented computing (SOC) [22] aims at building a complex system by composing less complex, loosely coupled building blocks called *services*. A service is an autonomous system providing its functionality via a well-defined interface. This interface is used to communicate with other services. Consequently, the *composition* of services into a new service is a key feature of SOC. Functional and non-functional *correctness* of a service composition is critical [27]. We define functional correctness in terms of the composition's *behavior*, for example deadlock freedom or weak termination. Non-functional properties refer to *Quality of Service* (QoS) dimensions like duration, response time, capacity, or reliability [7].

Time is an abstract concept which is crucial for many real-world systems like workflow systems [4], web services [3], or any kind of protocol [24]. The introduction of time to SOC affects both the behavior and the QoS of a composition [15,28]. On the one hand, timed constraints can limit the behavior of a composition. As an example, consider an airline booking service where a travel agency service may reserve a ticket for at most one hour. A travel agency service that always attempts to buy a ticket one day after reservation will always encounter a timeout. Hence, the behavior of the composition of these two services is limited to unsuccessful buying attempts. On the other hand, a composition may have to additionally satisfy timed requirements. For example, a travel agency service may prefer an airline booking service that takes less time to perform a reservation than other functionally equivalent airline booking services. In this paper, we investigate timed behavior and QoS of timed services, resp. of a timed service composition.

Given the importance and influence of time for services, there exist many different problem statements and approaches in the available literature. This makes it difficult to gather a common view on timed services. This survey elaborates on the problem

statements found in literature and gives an overview of state-of-the-art approaches. More precisely, the main contributions of this paper are as follows. The first step consists of defining a *problem space* — that is, a set of possible problem statements related to timed services. We present a problem space building upon five orthogonal problem dimensions in Sect. 2. In the second step, we *survey* a selection of existing approaches according to their corresponding problem statement in Sect. 3. Finally, we *evaluate* the state-of-the-art for every problem statement: "Is it valid and relevant?" "Is it an open problem?" "If it is solved, what is the quality of the applied approaches?". Section 4 concludes the paper with a discussion of future work and a conclusion.

2 Problem space

Most approaches for timed services deal with a specific problem statement. Instead of classifying the approaches directly, we start by classifying the problem statements into a problem space. This way, we gain a systematic overview on timed services: While a classification of the approaches may deliver interesting results on its own, a classification of the problem statements helps identifying problem classes that require further research.

Our problem space is a systematic collection of time-related challenges that arise in SOC. Every point in the problem space represents a unique combination of certain problem *characteristics*. It is spanned by five orthogonal *dimensions*: criterion, lifecycle phase, time abstraction, timed constraint, and system (see Fig. 1 for an overview). The selected dimensions and their characteristics are tailored towards the properties of the problem statements that arise when timed services are considered.

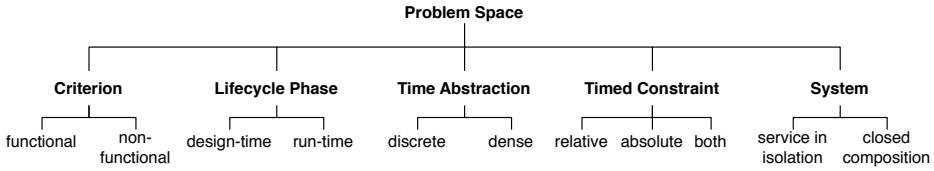

Fig. 1. Problem dimensions and their characteristics

In the following, we explain each dimension and its possible characteristics in more detail. For each, we give a description illustrated with some brief examples, explain its origin, and justify its relevance for timed services.

Criterion When dealing with timed services, two types of correctness criteria are relevant for a service-oriented system: functional and non-functional correctness. *Functional correctness* corresponds to the question whether the system works correctly — that is, the system's behavior. It covers for example deadlock freedom, weak termination, or the satisfaction of a set of timed requirements. Functional correctness is a precondition for *non-functional correctness*, which corresponds to qualitative questions — that is, how well does the system work. Examples for non-functional correctness are Quality of Service criteria — that is, duration, response time, capacity (messages per time unit), reliability, or price.

Lifecycle phase A lifecycle is a structure consisting of distinguishable phases that is imposed on the development of a system. In this paper, we consider design-time and run-time as characteristics. *Design-time* is the phase where the services and compositions are defined and implemented. Here we consider preconceived compositions, models and static service definitions. *Run-time* is the phase during which the system is executed. At run-time, we may have different information at our disposal, leading to new problems like monitoring, run-time adaption, instance migration, or re-configuration.

The aforementioned problem dimensions — criterion and lifecycle phase — are ubiquitous and inevitable, and concern any kind of information system [25,10].

Time abstraction There are two ways of introducing time into a system, distinguishable by a different resolution. *Discrete time* uses a domain with countably many time values, while *dense time* uses a domain with uncountably many time values. A problem which is decidable in discrete time may become undecidable when regarded with dense time.

Timed constraint We have two combinable ways of expressing timed constraints. Firstly, timed constraints can be *relative* to some event like the occurrence of an action or the entry of a state. Secondly, timed constraints can be *absolute* — that is, referring to a commonly known point in time.

The problem dimensions time abstraction and timed constraint are inherent to any kind of information system dealing with a notion of time. However, the last dimension — system — is a direct result of our service-oriented setting.

System In SOC we distinguish between *open* and *closed* systems — that is, between a service in isolation and a closed composition. Both systems elicit fundamentally different questions. For a service in isolation, questions like well-formedness or controllability arise. By contrast, questioning for example deadlock freedom is meaningful for a closed composition only.

The combination of all problem characteristics yields 48 different classes. In the next section, we present the survey and classify all examined approaches for timed services according the defined problem space.

3 Survey

We conduct the survey following a three-step approach by Levy and Ellis [20]: Firstly, we query Google Scholar[1], Mendeley[2], DBLP[3], and IEEE Xplore[4] by keywords (“service”, “soc”, “time”, “timing”, “realtime”, “timed constraints”, “timed requirements”, “quality of service”), and conduct both backward and forward search on the found literature. This yields a total of 55 papers, 26 of which we find relevant to this survey. Secondly, we analyze the approaches related to timed services. Finally, we classify them according to their problem statements; see Table 1 for the result.

[1] `http://scholar.google.de/` [2] `http://www.mendeley.com/`
[3] `http://dblp.uni-trier.de/` [4] `http://ieeexplore.ieee.org`

We now give a brief overview of the surveyed approaches, which we group into approaches focusing on functional correctness and approaches dealing with non-functional correctness. Later, we summarize our findings.

3.1 Functional correctness

Most of the surveyed approaches focus on a service or a composition specified in BPEL. The general approach consists of providing BPEL with a formal semantics for verification purposes.

Mateescu et al. [21] propose a translation of BPEL to discrete-time labelled transition systems which handle activity durations and timeouts. Timed safety and liveness properties are analyzed using model checking tools.

Kazhamiakin et al. [19] use web service timed transition systems (WSTTS) for the analysis and verification of timing aspects of BPEL4WS compositions. WSTTS are a variant of timed automata tailored towards web services. All time-related constructs of BPEL4WS incl. absolute timeouts are supported. Timed requirements can be expressed using a discrete subset of duration calculus. Additionally, the authors present an algorithm to compute extremal bounds of the execution duration of a composition.

Guermouche et al. [14] propose the FIACRE verification language as their underlying formalism. It supports a rich set of timed constraints (activity durations, message delays and timeouts) which is further separated into local and global constraints. Due to this distinction, the authors can formulate a well-formedness criterion for isolated services and a compatibility criterion for service compositions. Furthermore, their approach supports rich timed requirements using a timed leads to operator.

Similarly, Fares et al. [13] capture both the behavioral and the timing aspects of all BPEL 2.0 constructs by mapping them to FIACRE. Timed requirements can be formulated as LTL formulas and are verified against a service composition using the ToolBox Tina.

Kallel et al. [18] employ XTUS-automata for the specification and verification of relative and absolute timed constraints. The authors propose to use existing model checking tools to verify functional correctness criteria such as deadlock freedom. In addition, they present a translation of the formal specifications to AO4BPEL aspects which enforce the temporal constraints at run-time.

The approach proposed by Song et al. [26] aims to verify timed requirements for a BPEL composition with the goal of identifying other services suitable for composition at run-time. It is based on a mapping to Time Petri-nets and an algorithm to compute the extremal bounds of the time interval between two transitions. The approach only supports the specification of simple timed requirements on a time interval between transitions.

Haddad et al. [15,16] treat functional correctness of isolated services. They introduce an algorithm that either generates a correct interaction controller for a given BPEL specification or detects whether the specification is ambiguous. The absence of ambiguity can be regarded as a correctness criterion for isolated services.

Benatallah et al. [2] describe an extension for business protocols with timeouts on the states of a protocol. Based on these timed business protocols, they formulate the notions of time-dependent compatibility and replaceability.

Ponge et al. [23,24] extend this approach by introducing richer timed constraints and fine-grained classes of time dependent compatibility and replaceability properties. These classes are characterized by a set of operators that manipulate and analyze timed protocols. A mapping from timed protocols to a special class of timed automata allows the authors to derive decidability results for these operators. These results come at the cost of requiring deterministic service behavior.

The approach presented by Berardi et al. [3] employs timed finite state automata to represent two types of relative timed constraints (timeouts and durations). Again, deterministic behavior is a requirement for the services.

Čaušević et al. [9] extend the resource-aware, timed hierarchical language REMES for behavioral service modeling. They focus on service capacity and time-to-serve as timed requirements. The correctness of a service composition can be verified by employing Dijkstra's and Scholten's strongest postcondition semantics. The approach is limited to synchronous communication between the services.

Zahoor et al. [29] introduce a declarative approach for modeling web services based on event calculus. Given the composition design with a timed properties representation, an event calculus reasoner can be used to compute a solution satisfying associated timed properties. The approach supports synchronous and asynchronous communication.

de Alfaro et al. [12] present an approach to check the compatibility of timed interfaces. A timed interface is a specification of the input assumptions and output guarantees (incl. timing) of a component. The authors develop a well-formedness and a compatibility criterion for such timed interfaces and present algorithms to decide these properties.

Based on this work, Henzinger et al. [17] present an interface algebra for real-time components. Here, an interface specifies guaranteed task latencies depending on assumptions about task arrival rates and allocated resource capacities. Interface compatibility can be checked on partial designs. An interface is comparable to a stateless service.

3.2 Non-functional correctness

In the area of non-functional correctness, there are several approaches that deal with the computation and optimization of the QoS of a service composition at runtime. Cardoso et al. [7] propose a predictive QoS model that allows the computation of the QoS of a workflow from the QoS values of the tasks. Task QoS values are updated at runtime by monitoring the execution of the workflow, and the approach uses probability estimates for workflow transitions during the computation of the overall QoS. Many of the other approaches in the area cite this model or propose a similar one. Zhao et al. [31] also present a QoS model that allows the computation of overall QoS of a service choreography specified in the Chor language.

Based on the assumption that several functionally equivalent services exist for each activity of the workflow, the QoS of a service composition can be optimized by computing an optimal selection of participating services. Zeng et al., Canfora et al. and Aggarwal et al. all propose similar approaches which only differ in the used QoS model and optimization method [1,6,30]. Canfora et al. [5] present a method to handle expected QoS violations by replacing services during the execution of a service composition. Cardoso et al. [8] propose an approach to include QoS values to select services that do not violate the QoS requirements of a composition.

3.3 Findings

Table 1 presents the classification of the surveyed approaches according to their respective problem statements. Some approaches show up multiple times because they attack more than one problem class. Notice that any approach which supports dense time naturally also supports discrete time. However, in such cases a discrete time approach may exist which has a lower computational complexity. Hence, we do not list dense time approaches in the row of discrete time approaches. Only three approaches deal with both absolute and relative timed constraints [13,18,19]. Since every other approach is limited to relative constraints, we omit the timed constraint dimension in Table 1.

		functional		non-functional	
		open	closed	open	closed
discrete	design-time	[21]	[3], [23], [2], [29], [19]		
	run-time		[29]		
dense	design-time	[16], [15], [12], [14], [17]	[12], [24], [14], [18], [9], [13]	[7], [31]	[7], [31]
	run-time		[26]	[30], [6], [1], [5], [8]	[18], [30], [6], [1], [5], [8], [11]

Table 1. Classification of the surveyed approaches

There are two clusters of approaches. Each cluster represents a problem which has been treated by several authors with different formalisms. The largest cluster of approaches deals with ensuring functional correctness of a service composition at design-time. Most of the approaches in this cluster use dense time. Naturally, the approaches vary in their expressiveness, but this is outside the scope of this paper. It should also be noted that some approaches consider deterministic services only [2,3,12,23,24].

The second significant cluster is in the area of ensuring non-functional correctness at run-time. Most authors propose similar approaches to optimize the overall QoS of a workflow by computing an optimal selection of services. The underlying assumption is that for each activity of the workflow there exist several functionally equivalent services. They solve the resulting optimization problem with methods like linear programming [1,30] or genetic algorithms [5,6]. Other approaches in this cluster deal with monitoring temporal QoS constraints [18] or semantic service composition [8].

There are several sparsely populated spots in the problem space. Only two approaches deal with functional correctness at run-time [26,29], both with limited expressiveness. This may be due to the fact that the computational complexity of all approaches that attack functional correctness is so high that their application at run-time is not feasible in realistic scenarios [26].

Another area where we found little to no existing research is the problem of verifying non-functional correctness at design-time. It can be argued that reasoning about non-functional correctness criteria at design time is not very useful because QoS attributes of service must be constantly updated and monitored at run-time and thus an optimal selection of services for a composition can only be computed at run-time.

4 Discussion and Conclusion

In this paper, we survey approaches for timed services. We define a problem space with five dimensions and classify the surveyed approaches according to this problem space. Thereby, we identify which problems are already thoroughly researched and which problems warrant further research.

It could be debated that our problem space is inaccurate and that we should include more dimensions. Indeed, it would be interesting to differentiate the surveyed approaches further by their chosen formalism(s) or verification techniques. Nonetheless, such features are not related to the problem space and should therefore be analyzed separately. In any case, this could not be included in this paper due to space limitations.

Currently, we distinguish only between design-time and run-time. These phases of a traditional software lifecycle may be insufficient to describe the lifecycle of a service-oriented system. We are going to investigate whether the lookup and composition phase should be separated from the run-time phase.

We identify two areas in the domain of timed services which warrant further research: functional correctness at run-time and non-functional correctness at design-time. We intend to focus on the former problem class in the future. Hence, our future work is divided into three areas. Firstly, we will extend this survey by including more approaches and by classifying the approaches by their features — that is, their formalism and verification technique. Secondly, we will research correctness criteria for isolated timed services. We are particularly interested in the properties of well-formedness and controllability of non-deterministic, asynchronously communicating timed services. Thirdly, we plan to investigate methods for verifying functional correctness at run-time, specifically during the composition phase we mentioned above. Due to the dynamic nature of service-oriented systems, it is not sufficient to ensure functional correctness at design time. However, the existing approaches cannot be easily employed at run-time because of their computational complexity.

References

1. Aggarwal, R., Verma, K., Miller, J., Milnor, W.: Constraint Driven Web Service Composition in METEOR-S. In: IEEE International Conference on Services Computing (2004)
2. Benatallah, B., Casati, F., Ponge, J., Toumani, F.: On Temporal Abstractions of Web Service Protocols. In: Proceedings of the CAiSE Forum (2005)
3. Berardi, D., De Rosa, F., De Santis, L., Mecelia, M.: Finite state automata as conceptual model for e-services. Contract (2003)
4. Bettini, C., Wang, X., Jajodia, S.: Temporal reasoning in workflow systems. Distributed and Parallel Databases 11(3) (2002)
5. Canfora, G., Di Penta, M., Esposito, R., Villani, M.L.: QoS-Aware Replanning of Composite Web Services. IEEE (2005)
6. Canfora, G., Penta, M.D., Esposito, R., Villani, M.L.: An approach for QoS-aware service composition based on genetic algorithms. Genetic And Evolutionary Computation Conference (2005)
7. Cardoso, J., Miller, J., Sheth, A., Arnold, J.: Modeling Quality of Service for Workflows and Web Service Processes. Tech. rep. (2004)

8. Cardoso, J., Sheth, A.: Semantic E-Workflow Composition. Journal of Intelligent Information Systems (2003)
9. Causevic, A., Seceleanu, C., Pettersson, P.: Modeling and Reasoning about Service Behaviors and Their Compositions. Lecture Notes in Computer Science (2010)
10. Chung, L., do Prado Leite, J.: On non-functional requirements in software engineering. Conceptual modeling: Foundations and applications (2009)
11. Cristian, F., Fetzer, C.: The timed asynchronous distributed system model. Parallel and Distributed Systems, IEEE Transactions on (1999)
12. De Alfaro, L., Henzinger, T.A., Stoelinga, M.: Timed Interfaces. International Conference on Embedded Software (2002)
13. Fares, E., Bodeveix, J.P., Filali, M.: Verification of Timed BPEL 2.0 Models. Tech. rep. (2011)
14. Guermouche, N., Zilio, S.D.: Formal Requirement Verification for Timed Choreographies. Tech. rep. (2011)
15. Haddad, S., Melliti, T., Moreaux, P.: Modelling web services interoperability. International Conference on Enterprise Information Systems (2004)
16. Haddad, S., Moreaux, P., Rampacek, S.: Client Synthesis for Web Services by way of a Timed Semantics. International Conference on Enterprise Information Systems (2006)
17. Henzinger, T.A., Matic, S.: An Interface Algebra for Real-Time Components. IEEE RealTime and Embedded Technology and Applications Symposium (2006)
18. Kallel, S., Charfi, A., Dinkelaker, T., Mezini, M., Jmaiel, M.: Specifying and Monitoring Temporal Properties in Web Services Compositions. IEEE European Conference on Web Services (2009)
19. Kazhamiakin, R., Pandya, P., Pistore, M.: Representation, verification, and computation of timed properties in web service compositions. IEEE International Conference on Web Services (2006)
20. Levy, Y., Ellis, T.J.: A systems approach to conduct an effective literature review in support of information systems research. Informing Science Journal 9 (2006)
21. Mateescu, R., Rampacek, S.: Formal modelling and discrete-time analysis of BPEL web services. International Journal of Simulation and Process Modelling (2008)
22. Papazoglou, M.: Web Services - Principles and Technology. Prentice Hall (2008)
23. Ponge, J., Benatallah, B., Casati, F., Toumani, F.: Fine-grained compatibility and replaceability analysis of timed web service protocols. In: International Conference on Conceptual Modeling (2007)
24. Ponge, J., Benatallah, B., Casati, F., Toumani, F.: Analysis and applications of timed service protocols. ACM Transactions on Software Engineering and Methodology (2010)
25. Ramamoorthy, C.V., Prakash, A., Tsai, W.T., Usuda, Y.: Software Engineering: Problems and Perspectives. Computer (1985)
26. Song, W., Ma, X., Ye, C., Dou, W., Lü, J.: Timed Modeling and Verification of BPEL Processes Using Time Petri Nets. International Conference on Quality Software (2009)
27. Ter Beek, M., Bucchiarone, A., Gnesi, S.: Formal methods for service composition. Annals of Mathematics, Computing & Teleinformatics (2007)
28. Tsai, W.T., Lee, Y.h., Cao, Z., Chen, Y., Xiao, B.: RTSOA: Real-Time Service-Oriented Architecture. IEEE International Symposium on ServiceOriented System Engineering (2006)
29. Zahoor, E., Perrin, O., Godart, C.: A declarative approach to timed-properties aware Web services composition. Event (2010)
30. Zeng, L., Benatallah, B., Dumas, M., Kalagnanam, J., Sheng, Q.Z.: Quality driven web services composition. International World Wide Web Conference (2003)
31. Zhao, X., Cai, C., Yang, H., Qiu, Z.: A QoS View of Web Service Choreography. IEEE International Conference on e-Business Engineering (2007)