A Scenario-Based Governance Method for Coordination of Service Life Cycles

Sietse Overbeek, Marijn Janssen, Yao-Hua Tan

A Scenario-based Governance Method for Coordination of Service Life Cycles

Sietse Overbeek, Marijn Janssen, and Yao-Hua Tan

Faculty of Technology, Policy and Management,
Delft University of Technology,
Jaffalaan 5, 2628 BX Delft, The Netherlands, EU
{S.J.Overbeek,M.F.W.H.A.Janssen,Y.Tan}@tudelft.nl

Abstract. Most of today's organizations are still far from profiting from the full potential of web service technology. Organizations invoke each other's web services, but do hardly synchronize the life cycles of individual web services with each other. When realizing an integrated service this might cause failures and requires a governance method for synchronizing the life cycles among organizations. In this paper, we emphasize that there is a need for such a method and the basis of the method is explained. This consists of the identification of possible life cycle related scenarios when trying to integrate services. The method also provides support to coordinate and track changes in service life cycles. Based on the different scenarios when attempting to integrate services that have different life cycles, coordination to communicate life cycle changes and having clear agreements about expectations are needed for successful service integration.

1 Introduction

Web service technology is becoming the solution to make the business services of single organizations and inter-organizational coalitions available online and to match the supply of services and the demand for services by clients (see e.g. [5]). A web service can be defined as a software component identified by a URI, whose interfaces and bindings can be defined, described and discovered as XML artifacts [3]. However, most of today's organizations are still far from profiting from the full potential of web service technology consisting of the computerization, integration and matching of services. In fact, it is common practice for most organizations to develop their web services by adding a thin SOAP / WSDL / UDDI layer on top of existing software applications or components [5]. While simple services may be constructed this way, it is not sufficient for realizing and offering a dynamically created integrated web service that matches complex and varying client needs. The notion of a *service-oriented life cycle methodology* has been introduced to design, implement, monitor, and manage web services in such a way that organizations can benefit in full from the advantages of web services. Such methodologies also provide sufficient principles and guidelines to specify, construct, refine and customize highly flexible business processes taken from a

set of internal and external web services [4]. This enables that the specification and execution of business processes is aligned with those business services that are transformed to web services. Examples of existing methodologies can be found in [1, 4]. In practice, organizations invoke each other's services and become dependent of those services. Despite these dependencies, organizations adopt or create their own methodologies which are often unrelated to those of other organizations. This increases the risk that the various life cycles are out of sync which might result in failures.

This is a complicating factor when integrating web services and the accompanying business processes that need to be executed to supply the integrated service. In the context of supply chain logistics, for example, an integrated web service can be supplied to a client who wishes to declare veterinary cargo online and to track and trace that type of cargo. If the integrated web service is a new web service that is going to be offered by two different organizations then their life cycle methodologies should be applied synchronously. This will prevent, e.g., communication problems and delays in the joint development of the integrated service and the accompanying cross-organizational business process. Mismatches between life cycles can also occur if, for example, an integrated web service needs to contain both existing and new web services. A service that already exists is in a different phase of its life cycle than a newly created service. Moreover, organizations that collaborate regularly innovate their processes, methods and business models and in this way they are in need for substituting old services with new ones. In [6], the need for these innovations are also linked to those organizations that rapidly expand. Innovation causes organizations to phase out old services and add new services. This implies again that these services are in different phases of their life cycles. In this paper, we present a governance method for supporting the coordination of web service life cycles. The aim of the method is to ensure that the dependencies among web services from different organizations are managed during the complete life cycle. This scenario-based method provides a way of working for service providers in coordinating and keeping track of life cycle changes. Section 2 introduces four different scenarios that can occur when attempting to integrate web services that have different life cycles. Subsequently, the basis for the proposed governance method is presented in section 3. Finally, the conclusions of this paper are presented in section 4.

2 Scenarios for comparison of service life cycles

In order to understand how web services and their life cycles can be compared and to determine to what extent they match, we formalize four possible matching scenarios. A life cycle methodology is typically divided in various phases to depict in which position a web service is in its life cycle. The phase equation is used to determine which phases uniquely belong to which life cycle methodology: $\mathsf{Phase} : \mathcal{PS} \rightarrow \mathcal{LC}$. If a phase $p \in \mathcal{PS}$ is part of a life cycle methodology $l \in \mathcal{LC}$, this can be expressed as $\mathsf{Phase}(p) = l$. In this case, the set $\mathcal{PS}$ is the set of phases and $\mathcal{LC}$ is the set of service-oriented life cycle methodologies. The phase

classification equation is used to determine in which phase of its life cycle a web service is classified: $\mathsf{Classification} : \mathcal{SC} \rightarrow \mathcal{PS}$. For example, if a service $s \in \mathcal{SC}$ is in phase $p \in \mathcal{PS}$ of its life cycle, this can be expressed as $\mathsf{Classification}(s) = p$. In this case, the set $\mathcal{SC}$ is the set of services. When different life cycle methodologies are used, it may be impossible to compare two web services that have life cycles that are based on different methodologies. This is because different terms to describe a phase are used and web services can be classified in a certain phase based on different criteria. However, comparison may still be possible if some phases of each life cycle are semantically similar. For example, a 'planning' phase in one life cycle may have the same meaning as an 'initiation' phase in another life cycle. Semantic similarity between phases of life cycle methodologies is modeled as follows: $\mathsf{Similarity} : \mathcal{PS} \times \mathcal{PS} \rightarrow [0, 1]$. The lack of semantic similarity between two phases $p_1, p_2 \in \mathcal{PS}$ can be expressed as $\mathsf{Similarity}(p_1, p_2) = 0$, while the opposite result is true for full semantic similarity between two different phases. Four scenarios are imaginable when matching life cycles: (1) Two different services are in the same phase of their life cycles and the life cycles are also applications of the same methodology. (2) Two different services are in different phases of their life cycles and the life cycles are again applications of the same methodology. (3) Two different services are in different, but semantically similar phases of their life cycles that are based on two different methodologies. (4) Two different services are in semantically distinct phases of life cycles that are based on two different methodologies. These scenarios can be described more formally by using the above equations:

$$\exists_{l \in \mathcal{LC}} \exists_{p \in \mathcal{PS}} \exists_{s_1, s_2 \in \mathcal{SC}} [\mathsf{Phase}(p) = l \wedge \mathsf{Classification}(s_1) = p \wedge \mathsf{Classification}(s_2) = p] \quad (1)$$

$$\exists_{l \in \mathcal{LC}} \exists_{s_1, s_2 \in \mathcal{SC}} [\mathsf{Phase}(\mathsf{Classification}(s_1)) = l \wedge \mathsf{Phase}(\mathsf{Classification}(s_2)) = l \wedge \mathsf{Classification}(s_1) \neq \mathsf{Classification}(s_2)] \quad (2)$$

$$\exists_{l_1, l_2 \in \mathcal{LC}} \exists_{s_1, s_2 \in \mathcal{SC}} [\mathsf{Phase}(\mathsf{Classification}(s_1)) = l_1 \wedge \mathsf{Phase}(\mathsf{Classification}(s_2)) = l_2 \wedge \mathsf{Classification}(s_1) \neq \mathsf{Classification}(s_2) \wedge \mathsf{Similarity}(\mathsf{Classification}(s_1), \mathsf{Classification}(s_2)) = 1] \quad (3)$$

$$\exists_{l_1, l_2 \in \mathcal{LC}} \exists_{s_1, s_2 \in \mathcal{SC}} [\mathsf{Phase}(\mathsf{Classification}(s_1)) = l_1 \wedge \mathsf{Phase}(\mathsf{Classification}(s_2)) = l_2 \wedge \mathsf{Classification}(s_1) \neq \mathsf{Classification}(s_2) \wedge \mathsf{Similarity}(\mathsf{Classification}(s_1), \mathsf{Classification}(s_2)) = 0] \quad (4)$$

Under normal circumstances, it can be expected that in the first matching scenario the least difficulties exist to integrate web services and that these difficulties will gradually increase up until the fourth scenario. If this assumption is true, this would mean that there is a causal relation between the life cycles of web services and the ability to integrate and supply them as one integrated service. Examples of causes for integration difficulties are: disabilities to comprehend service designs, conflicting service designs, temporal differences between activities performed in comparable phases and different service maintenance levels.

3 Scenario-based governance method for life cycle coordination

The proposed governance method is based on the underlying thought that *changes* in the life cycles of web services require proper coordination, just like the matching of supply and demand of services should be coordinated [2]. This will also include the final agreements on these changes by the different owners of the web services. Changes in life cycles need to be announced and the time-line to make actual changes to the services need to be agreed on. A governance method that can be used by service providers as a *way of working* to keep track of desired changes and to coordinate them to ensure the proper functioning of an integrated service is then called for. The governance method consists of two parts, which concerns the activity diagrams shown in figures 1 and 2. The first diagram shows how to determine which of the four presented scenarios apply when attempting to realize an integrated service. To determine which scenario can be considered,

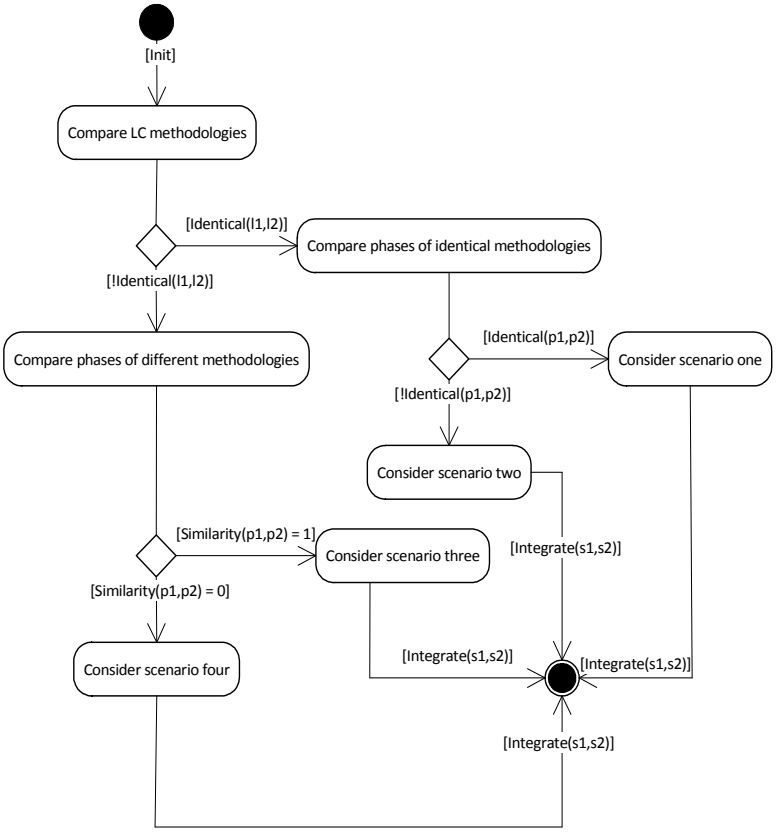

Fig. 1. Determining life cycle scenario when realizing an integrated service.

the life cycle methodologies of the services that would be part of a new integrated service are compared. If the methodologies are identical, the phases of the service life cycles can be compared. If those phases are also identical, scenario one can be considered. If not, the life cycles resemble scenario two. If the methodologies are not identical, it should be determined whether the life cycle phases are semantically similar or not. If this is the case, we arrive at scenario three. If it is not the case, we arrive at scenario four. The integration procedure can be started after determining with which scenario is dealt with and taking into account the possible issues that can arise related to that scenario. When the proper scenario is determined, service providers are more aware of which possible issues they may face during the integration process. The second diagram shows how to coordinate and keep track of changes in service life cycles. A change in a

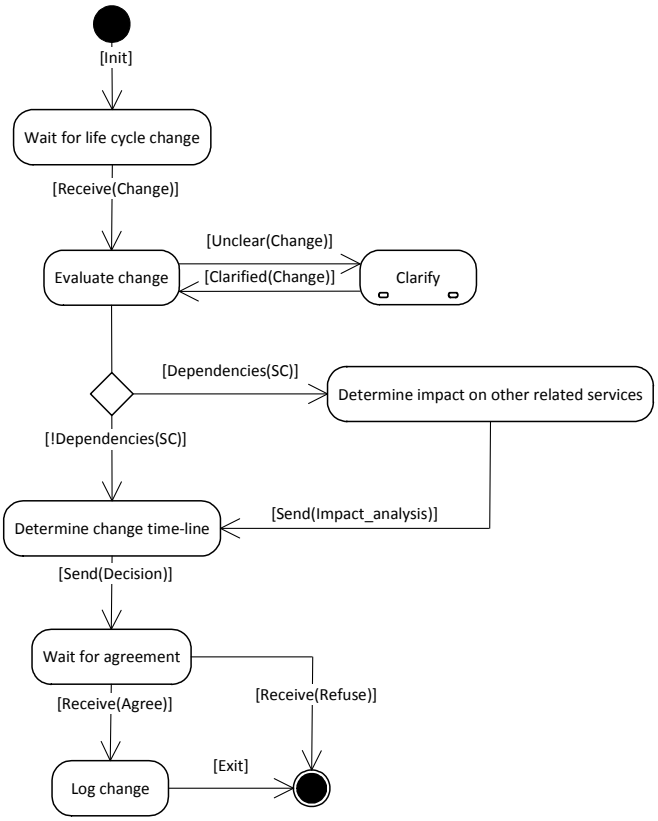

Fig. 2. Coordination and tracking of changes in service life cycles.

life cycle of a service that is known to a service provider will be evaluated after its reception. More clarification concerning the change can be requested if this is needed. The clarify state represents a composite state, which is not shown

further. The impacts of the change on other services are determined if there are dependencies between the service of which the life cycle has changed with other services. Next, the time-line of the change is determined. After determining this, the change can be either agreed or refused. The change is logged in a registry of life cycle changes if it is agreed upon. Both the scenario determination part and the part to coordinate life cycle changes can be used by service providers to control attempts to realize integrated services.

4 Conclusions

The results of the presented research show the basis of a governance method for the coordination of web service life cycles. The motivation to create such a method is rooted in the observation that organizations invoke each other's web services when realizing an integrated service for their clients, but that the life cycles of those services are hardly synchronized. The actual governance method consists of two parts for supporting the coordination of life cycles. One part shows how to determine with which life cycle scenario a service provider is confronted when integrating services. The second part provides a way of working for service providers in coordinating and keeping track of life cycle changes. By adopting the governance method, coordination to communicate changes and having clear agreements about expectations is then realized based on the different scenarios when attempting to integrate services that have different life cycles.

References

1. Bianchini, D., Cappiello, C., De Antonellis, V., Pernici, B.: P2S: A methodology to enable inter-organizational process design through web services. In: van Eck, P., Gordijn, J., Wieringa, R. (eds.) 21th International Conference on Advanced Information Systems Engineering, CAiSE 2009, Amsterdam, The Netherlands, June 8 - 12, 2009, Proceedings. Lecture Notes in Computer Science, vol. 5565, pp. 334–348. Springer, Berlin, Germany, EU (2009)
2. Burstein, M., Bussler, C., Zaremba, M., Finin, T., Huhns, M., Paolucci, M., Sheth, A., Williams, S.: A semantic web services architecture. IEEE Internet Computing 9(5), 72–81 (2005)
3. Ferris, C., Farrell, J.: What are web services? Communications of the ACM 46(6), 31–34 (2003)
4. Papazoglou, M., van den Heuvel, W.J.: Service-oriented design and development methodology. International Journal of Web Engineering and Technology 2(4), 412–442 (2006)
5. Papazoglou, M., van den Heuvel, W.J.: Business process development life cycle methodology. Communications of the ACM 50(10), 79–85 (2007)
6. van de Weerd, I., Brinkkemper, S., Versendaal, J.: Incremental method evolution in global software product management: A retrospective case study. Information and Software Technology 52(7), 720–732 (2010)