Analysis of Message Mismatches Solutions in WSMO

Kanmani Munusamy[a,]*, Suhaimi Ibrahim[b], Norbik Bashah Idris[b], Harihodin Selamat

[a]*Department of IT Services,University Malaya (UM),50603, Kuala Lumpur,Malaysia*
[b]*Advanced Informatics School (AIS), Universiti Teknologi Malaysia (UTM),54100 Kuala Lumpur, Malaysia*

Abstract

Mediating between web services has been widely discussed in Service Oriented Architecture and during composition of web service. The existing researches have addressed various techniques that able to mediate the process and data flow between the web services. These approaches are highly dependent on the developer during data mapping and modeling of the process mediators at design time. Therefore, resolving message mismatches in Semantic Web Service (SWS) such as Web Service Modeling Ontology (WSMO) has been emerged in order to increase the degree of automation in this task. However, we find that the existing knowledge representations of the web service using ontology and the transition rules in the choreography interface are not sufficient in order to generate the process mediator automatically.

Keywords: SEMANTIC WEB SERVICE, PROCESS MEDIATION, MESSAGE MISMATCHES

1. Introduction

Generally, capability of an offered service needs to be matched with the goal of the requested service from various perspectives in order to discover and select the matched services. However, there is no assurance that these matched services can work well since it may end at a deadlock which only known during the actual invocation phase. Thus, process mediation which focuses on the behavioral and communication mismatches during web service interaction is essential in ensuring compatibility of the web services. The process mediation can be visualized as a middle service that ensures two web services during choreography (interaction between service requestor and provider) or orchestration (interaction between provided services) are able to interact successfully.

During the discovery and selection phases, the capability of an offered service will meet with the goal of the requested service from various perspectives in order to discover and select the services that are compatible as explained in [1]. However, there is no assurance that these matched services can interact well since they may meet with a deadlock which can only be detected during the actual invocation phase. Behavioral incompatibility in the selected web services can lead to the termination of a web service

* Corresponding author. Tel.: +06-012-7607400; fax: +06-03-26930933.
E-mail address:kanmani@um.edu.my.

composition and invocation if they are not detected before the actual execution. Therefore it is important to identify behavioral compatibility between the web services and support the incompatibility using behavioral resolutions such as a mediator or an adaptor.

Behavioral compatibility is mainly concerned with mismatches in the web services interaction. It occurs when a web service is in use or when there's communication between several web services to produce the desired output based on the input received. It also ensures that business process of each individual web services executed correctly. Therefore, behavioral compatibility analysis closely related to business process and rule management techniques as explained in [2].The research on this behavioral incompatibility has been stated in various terminology such as mismatch in the Message Exchange Pattern (MEP) [3] or behavioral interface, process incompatibility [4] and protocol mismatches [5].

This paper is organized as follows: Section 2 describes the process mediation and provides definitions of process mediation and describes the types of mismatches. It also explains main elements that support process mediation. Section 3, describes process mediation approaches and the algorithm used in the WSMO Framework. Finally, Section 4 concludes the analysis process mediation in WSMO framework.

2. Mediation that Support Message Mismatches

Generally there are three type of message mismatches in a web service interaction namely, signature/data, functional and process. A signature level of mismatch refers to incompatibility in the data element of a web service with the data element of the interacting web service. The differences in structure, type and naming conventions of the data elements in web service messages can lead to behavioral incompatibility. Generally, the signature level of mismatches uses scheme mapping and data mapping methods to align the differences in the data elements.

It requires the involvement of the developer to identify the correct data mappings between the different web service applications. Next, the functional mismatches describe the differences in the offered and the requested functionalities between the service provider and the requestor. Both of the signature/data and functional level mismatches need to be combined in order to identify the subsequent, process level mismatches. Moreover, these levels of mismatches need to be supported by sufficient semantic in order to identify the actual meaning of the interacting messages depending on the web service domain.

The role of semantics in resolving signature/data is very essential to ensure the correctness of interaction between the web services. Ontology plays an important role in bringing the semantics in the behavioral analysis. The reasoning mechanism that supports ontology has become very useful in many automated tasks in web services such as discovery, selection and composition. Many Semantic Web Service Frameworks such as OWL-based web service (OWL-S) [6] , Web Service Modelling Ontology (WSMO) [7] and Semantic Annotations of WSDL (SAWSDL) have taken bottom-up approaches to bring the semantics into web services. These approaches specify the goal and capability of the web services semantically from the beginning. This bottom-up method is able to enhance the protocol and deadlock mismatch analysis through the available ontology domain.

The mediation terminology has been reintroduced in web services by [4]. They have described mediation as "*process for settling a dispute between two parties where a third one is employed whose task is try to find common ground that will resolve inconsistencies between their respective conceptualizations of a given domain*". Apart from the definitions of Fensel and Bussler, there are many other definitions for mediator in the context of web services. For instances, [8] define mediator as *software module that provides sharing of services and agglomeration of resources into complex service* and [3] define mediation *as ability to resolve heterogeneities among heterogeneous entities which allow*

parties to exchange messages, documents and data disrespect of their vocabulary and behavioral models. In this paper we have focused on data level message mismatches in existing semantic web service mediation solutions.

2.1. Type Of Message Mismatches

Fensel and Bussler [4] identified three types of communication mismatches between the Web Services, namely precise match, solvable and irresolvable mismatches. A precise match occurs when the sender web service sends the message in the exact order that the receiver web service has requested. Therefore it only requires data mediation to solve possible data or format mismatches. The unsolvable message mismatches usually comes to a dead end. This paper focuses on the solvable mismatches that have been highlighted by many researchers. There are five situations that generate solvable message mismatches as stated below:

- sender Web Service sends a message that is not expected by the receiver
- sender sends single message that is expected to be in multiple forms
- sender sends multiple messages that are expected to be in the single form
- sender sends messages in wrong order
- sender is not sending messages that are expected by the receiver.

Cimpian [9] has presented five ways that the process mediator could address solvable mismatches as listed below:

- it stops the original message since it is not requested by the receiver
- it splits the original message before reaching the receiver
- it combines the original message before it reaches the receiver
- it inverses the original messages before it reaches the receiver
- It sends a dummy message since a message is expected by the receiver

2.2. Data Mediation that Support Process Mediation

Most of the message mismatches approaches do not provide any explicit description of the data or the functional mediator due to two reasons. Firstly, it has been argued that generating data and the functional mediator can both be huge research topics on their own. Secondly, some approach gurus are under the assumption that the web services have compatible messages at the data and functional levels. Some approaches have adopted the data / functional component as external components or agents to support the process level mismatches. There a few available tools that support data mediators such as Microsoft Biztalk and Stylus Studio XML Schema Mapping. However, we do not come across any tools that support the functional mediator. Both the data and functional mediators play important roles in generating message mapping and identifying the correct message mismatches.

2.3. Elements that Supports Message Mismatches

There are four important elements that support message mismatches in semantic web services as discussed in [10] namely message mapping, identification of process mismatches, identification of the

mediator and the generation of the mediator. In this paper the first two elements that supports data level message will be discussed in detail.

2.3.1. Message Mapping

Message matching is an important step in producing a mediator. Behavioral compatibility between web services can be identified by the matching of messages that are involved in the interaction. Message matching involves data mapping (data mediation) and functional mapping/mediation in order to determine the similarity or differences in the operation name and input/output elements used in the web services. Generally the data and functional mapping only discusses implicitly in the process mediation approaches. There are few other issues are need to pay attention when handling the message mapping between the web services.

- *Semantic/syntactic support in order to generate message mapping*
  Semantic support is essential in order to provide automation in message mapping and improve the correctness of the mapping provided. As discussed earlier, most of the approaches assume that the operation and the parameter have identical names.
- *Elements that involved in the message mapping:*
  There are two types of mapping elements that have been widely used in the process mediation. Firstly using input and output of the related web service operatio, secondly using - the basic input and output of the operations are transformed into Input, Output, Precondition and Effect (IOPE) before identifying the message mapping
- *Rules that apply in message mapping:*
  Firstly, all the input requested from the provider needs to be delivered by the requestor; secondly, the preconditions of a state need to satisfy the requestor and finally all the output and effects that are requested by the requestor need to be provided by the provider.
- *Automation level in message mapping:*
  Most of the message mapping in the process mediator is provided by the developers who have sufficient knowledge on the domain of the specific scenario.

2.3.2. Identification of Process Level Mismatches

There are three other elements have been analyzed in identifying the protocol mismatches in web services as listed below namely, data and control flow analysis, the analysis method and the automation in identification of the protocol mismatches.

- *Data and control flow analysis:*

Data flow analysis helps the behavioral compatibility study based on the message exchanged between the web services. The messages exchanged between the provider and requestor of the web service is also identified as the operation that sends and receives between the target and the sourced web service. The control flow analysis ensures the process mediator supports the basic business logic in the web service interaction.

- *Analysis method:*

There are two main methods that have been adopted in order to identify the process mediator in the web service interaction namely the Specific Protocol Analysis (SPA) and the General Reachability Analysis (GRA). SPA method identifies each protocol mismatch individually and produced specific resolution for each of them. The GRA method determines whether the overall web service interaction is successful or not by ensuring there is no deadlock communications or information loss is exist in the interactions.

- *Automation in identification:*

Generally these process mediators are identified manually by the developer or automatically by using logic based algorithms or software tools.

3. Message Mismatches Handling in WSMO

This section present detail analysis on three approaches that uses WSMO framework in resolving message mismatches.

3.1.1. Mediator Component in WSMO

The WSMO framework provides a rich description of all the related aspects of Web Services through four important components which are: the goal, web service, ontology and the mediator. The ontology component plays an important role in this framework since it carries the semantic description for all the other components in this framework. The goal component defines the user's preferences with respect to the requested functionality and interfaces through the *requestedCapability* and the *requestedInterface*. On the other hand, the web service component defines the offered functionalities and the ways to interact with the services through the capability and interface elements.

Generally, the goal, the web service and the ontology components play a common role to bring the semantic description to a Web Service which is similar to other Semantic Web Services Frameworks. However, this framework has proposed a distinctive component known as the mediator to resolve the interoperability problems in Web Services at various levels such as data, functional and process mismatches. This component contains four elements which are the *OOMediator*, the *GGMediator*, the *WGMediator* and the *WWMediator* to overcome interoperability problems between different the WSMO components.

Based on the definition provided in [2], the *WGMediator* and the *WWMediator* are closely related to the process mediator. However, the implementation of the *WWMediator* in resolving these process mismatches is not specified clearly in any of the provided example.

3.1.2. Message based Process Mediation

One of the approach [11, 12], that implements in WSMO framework uses the Choreography Parser to check the mode of each state signature before it stores them in the Internal Repository. It matches the "in" mode instance in a goal/web service interface with the "out" mode instance from the web service/goal interface. This "in" and "out" list defines the data flow between the provider and the requestor. This data flow analysis is supported by the transition rule which defines the sequencing of the data exchange. This approach uses the WSML Reasoner to evaluate the transition rules before sending or deleting the stored instances.

However, this approach provides insufficient discussion on how choreography parser which matches the state signatures and the WSML Reasoner evaluates the transition rule that works together. Secondly, it also not utilizing the *WGMediator* or *WWMediator* components as explained in the WSMO concepts.

Thirdly, it only provides steps to resolve process mismatches in a specific scenario but not for the general algorithm as to how the process mediator could perform the transformation based on the state signatures.

3.1.3. Algorithm that supports Process

WSMO Framework also has been extended to resolve the mediation scenario in the SWS challenge [13]. In this approach, the message interactions are evaluated using both the transition rules and the data flow that are extracted from the choreography interface. Two important contributions of this approach have been extracted in order to resolve the process mediation.

Firstly, it has defined four basic choreography rules that are derived from the WSDL operations. The WSDL operations are classified into four patterns; in-out, in-only, out-only, out-in based on the sequence of input or output messages of the operation. Secondly, this approach provides a general algorithm that handles the communication mismatches.

Generally, this algorithm collects all the data that needs to be exchanged between the Web Services and store them into the memory. Firstly, it evaluates the transition rules in each Web Services and stores required actions such as add and remove into the memory. At the same time, it also sends the input parameter in each web service as stated in the choreography interface.

Secondly, it evaluates the action list in the memory and deletes the removed action and the corresponding data from the data list in the memory. It then checks the output symbols at each web service. The output symbols that are equivalent to the messages in the add action will be inserted into data list in the memory and removes the corresponding add action from the action list. This algorithm ensures that the each web service memory contains the expected incoming messages. Finally, it calls the data mediation to mediate the data in both Web Service memories.

This algorithm has addressed all mismatch patterns [9] except for the new generating messages. It allows message ordering and stops unexpected messages. The message merging and splitting is handled by the data mediation. This approach has provided a clear view on how the transition rules and the state signatures can be evaluated using an algorithm to generate the process mediation steps. Generally, this process mediation technique is similar to space based process mediator (SPM) [14] approach. It has provided memory space to the each web service to handle the data flow which is supported by the choreography transition rules. It does not specify the usage of the WSMO mediator components such as the *WGMediator* or *WWMediator* in the process mediation algorithm. Moreover, ability of the algorithm in handling complex mismatches or combination of more than one mismatch patterns are also underspecified.

4. Conclusion

For this paper, the important elements of data mediation which supports the process mediation in the Semantic Web Services were analysed. The analysis conducted on the approaches that have been collaborated with the WSMO framework. WSMO components that specifically support process mediation are discussed. This followed by the process mediation approaches that are related to the WSMO framework. Based on the analysis, we find that message mapping and identification of mismatches in process mediation solution in WSMO framework is highly depends on the developer. We also find the usage of semantics and reasoning very limited in these two specific elements of process mediation. Moreover, there is need in expending the current research in data mediation to understand the content of the web service messages.

Acknowledgements

This research is supported by Ministry of Higher Education Malaysia (MOHE) and Universiti Teknologi Malaysia (UTM) under Research University Grant (RUG) Vote Number 00H74.

References

[1] Keyvan, M., et al., *A comparative evaluation of semantic web service discovery approaches*, in *Proceedings of the 12th International Conference on Information Integration and Web-based Applications & Services*. 2010, ACM: Paris, France.

[2] Thirumaran.M, Dhavachelvan.P, and Thanigaivel.K, *Business Rule Management Framework for Enterprise Web Services.* International Journal on Web Service Computing (IJWSC), 2010. **1**(2): p. 15-29.

[3] Arroyo, S., M.A. Sicilia, and J.M. Dodero, *Choreography frameworks for business integration: Addressing heterogeneous semantics.* Computers in Industry, 2007. **58**(6): p. 487-503.

[4] Fensel, D. and C. Bussler, *The web service modeling framework WSMF*. Electronic Commerce Research and Applications, 2002. **1**(2): p. 113-137.

[5] Yellin, D.M. and R.E. Strom, *Protocol specifications and component adaptors.* ACM Transactions on Programming Languages and Systems, 1997. **19**(2): p. 292-333.

[6] Martin, D., et al. *OWL-S: Semantic Markup for Web Services* 2004 22 Nov 2004 [cited 28 June 2011]; Available from: http://www.w3.org/Submission/OWL-S/.

[7] Bruijn, J.d., et al. *Web Service Modeling Ontology (WSMO)*. 2005 3 June 2005 [cited 28 June 2011]; Available from: http://www.w3.org/Submission/WSMO/.

[8] Grahne, G. and V. Kiricenko, *Process Mediation in Extended Roman Model*, in *First International Workshop on Mediation in Semantic Web Services (MEDIATE 2005) conjuction with 3rd International Conference on Service-Oriented Computing (ICSOC 2005)*, M. Hepp, et al., Editors. 2005: Amsterdam, Netherlands p. 17-33.

[9] Cimpian, E. and A. Mocan, *WSMX process mediation based on choreographies*, in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. 2005: Nancy. p. 130-143.

[10] Munusamy, K., et al., *A Comparative Evaluation of State-of-Art Web Services to Support Protocol Mismatches.* International Journal on Web Service Computing (IJWSC), 2011. **2**(3).

[11] Cimpian, E., *Message-based Semantic Process Mediation*, in *Faculty Science*. 2010, National University of Ireland Galway: Ireland. p. 198.

[12] Cimpian, E. and A. Mocan. *Process Mediation in WSMX*. 2005 8 July 2005 [cited 28 June 2011]; Available from: http://www.wsmo.org/TR/d13/d13.7/v0.1/.

[13] Vitvar, T., et al., *Mediation using WSMO, WSML and WSMX*, in *Semantic Web Services Challenge*. 2009. p. 31-49.

[14] Zhou, Z., et al. *Developing process mediator for supporting mediated Web service interactions*. in *Proceedings of the 6th IEEE European Conference on Web Services, ECOWS'08*. 2008. Dublin.