Promoting B2B Integration with Semantic Web Service Technologies

Seyyed Ali Rokni Dezfooli  
PESE Laboratory, Sharif  
University of Technology,  
Tehran, Iran  
rokni@ce.sharif.edu

Jafar Habibi  
PESE Laboratory, Sharif  
University of Technology,  
Tehran, Iran  
jhabibi@sharif.edu

Soheil Hassas Yeganeh  
PESE Laboratory, Sharif  
University of Technology,  
Tehran, Iran  
yeganeh@mehr.sharif.edu

Abstract

Several e-business frameworks are developed to define standards for information sharing within and between companies. These frameworks only standardize structure of messages and aren’t able to define semantics. It leads to slow and costly dynamic Business-to-Business (B2B) integration. The use of Semantic Web Service (SWS) technologies has been suggested to enable more dynamic B2B integration of heterogeneous systems and partners. We present a semantic B2B mediator based on the WSMX—a SWS execution environment, to tackle heterogeneities in B2B services. We particularly show how WSMX can be made to support the RosettaNet e-business framework and how it can add dynamics to B2B interactions by automating mediation of heterogeneous messages. We used a reasoner in interaction with WSMX for automatic consistency checking of instances in runtime. The benefits of applying SWS technologies include more flexibility in accepting heterogeneity in B2B integrations and easing back-end integrations. This allows introducing more competition into the purchasing process within e-business frameworks.

1. Introduction

Several e-Business frameworks have been developed to integrate heterogeneous enterprise information systems [11, 13, and 15]. The e-business frameworks address Business-to-Business (B2B) integration on business process, message and communication levels [13]. Due to the flexibility of e-business frameworks regarding message details and message ordering, considerable effort is required to ensure that the B2B integration details of two partners, match. Thus traditional B2B integrations suffer from long implementation times and high costs, leading to long-term rigid partnerships [1, 2].

RosettaNet is a famous and widely used XML-based e-business framework that establishes a common language and standard processes for B2B transactions. It developed the Partner Interface Process (PIP) specification for public business processes between trading partners. The PIP standardizes public business process automation by standardizing business documents, the sequence of sending these documents, and the physical attributes of the messages that define the quality of service. The content of these PIPs are structurally validated by either DTD for the older and rarely used PIPs or XML Schema for the recently updated and usually used ones [3]. DTDs and XML Schemas lack expressive power to capture all necessary constraints and do not make all document semantics explicit. Being able to express constraints in machine interpretable format is expected by RosettaNet experts.

Semantic technologies and Semantic Web Services (SWS) have been proposed to achieve more dynamic partnerships [16]. Several SWS approaches such as OWL-S [5] or the Web Service Modelling Ontology (WSMO) [6] are offered to enabling semantic annotation of the B2B integration interfaces. This allows automated or semi-automated mediation.

The SWS solution proposed in this paper is based on the Web Service Modelling eXecution environment (WSMX) [14]. WSMX is a reference implementation of the Web Service Modelling Ontology (WSMO) and operates using the Web Service Modelling Language (WSML). WSMO is a meta-model for semantic web services related aspects.

The rest of paper is structured as follows: first we present related work to solving this problem in section 2. Then motivate our solution using an example in section 3. Section 4 presents the architecture of our semantic B2B mediator. Setup and runtime phase of integration is described in sections 5 and 6 respectively. In section 7 we explain expected benefit of our solution. We conclude our paper in section 8.
2. RELATED WORKS

There are a number of papers discussing the use of semantic web service to enhance current B2B standards. Some concentrate on ontologically annotated B2B standards [8, 10]. Foxvog and Bussler describe how EDI X12 can be presented using WSML, OWL and CycL ontology languages [10]. The paper focuses on the issues encountered when building general purpose B2B ontology, but does not provide an architecture or an implementation. Anicic et al. [9] present how two XML Schema-based automotive B2B standards are lifted using XSLT to OWL-based ontology. They use a two-phase setup and run-time approach similar to ours. The proposed approach in their works is based on different B2B standards and focus only on the lifting and lowering to the ontology level.

Others apply semantic technologies to B2B integrations [7, 8]. Preist et al. [8] presented a solution covering all phases of a B2B integration life-cycle. The paper addresses the lifting and lowering of RosettaNet XML messages to ontologies, but no richer knowledge is formalized or used on the ontological level. Trastour et al. [7, 8] augment RosettaNet PIPs with partner-specific DAML+OIL constraints and use agent technologies to automatically propose modifications if partners use messages differently. Their approach of accepting RosettaNet in its current form and lifting to semantic languages is similar to ours, but we go further by axiomatizing implicit knowledge and by providing mappings to resolve heterogeneity of data. In [4] a B2B gateway have introduced for solving heterogeneity problem. In comparison with other works, our approach uses an automatic way to generating ontology. Also, a reasoner used for consistency checking, of instances. In addition, modelled constraints validate that cover business constraints.

3. Motivation

Consider a marketplace with 5 organizations. We have shown this in figure 1.a. Black nodes support RosettaNet standards and white nodes have their proprietary point-to-point agreement. Assume all companies can communicate via SOAP over HTTP. Dashed lines are potential communications that in current situation are not active because of semantic problems. Since RosettaNet have some optional items in message detail and ordering, nodes 1 and 2 that have same standards, can’t communicate without ambiguity. In general, consider a marketplace with n organizations. These organizations have the potential of business interactions. Assume that some organizations produce similar products. It can lead to some can be competitor of some others.

![Figure 1. current and proposed scenario](image)

This marketplace can suffer from some problems:

- Because of B2B integration is very expensive; organizations interact only with several others that have pre-agreement with them. In sequence, B2B integration only covers purchasing activities and there is no competition for purchasing per delivery basis.
- Adding new member in this marketplace is expensive.
- Even in the case of some organizations that use the same B2B framework, pre-agreement is essential. Because of flexibility of e-business frameworks they have some optional items in message details and message ordering.
- If we want all organizations can communicate with others in a meaningful manner and without ambiguity, we need $O(n^2)$ pre-agreement or setup phase.

In order for integration to be possible, business partners must comply on three interoperability levels: Communication, message and business process. We focus on the two latter assuming both companies communicate via SOAP over HTTP. Message level interoperlation is the ability to understand content of exchanged messages. For example in figure 1.a, black nodes use PIP3A4 for create a purchase order request and confirmation message. Business Process level interoperation is the ability of companies to exchange messages in the right sequence and timing. In figure 1-a, black partners comply with PIP3A4 standard choreographies. We introduce a semantic B2B mediator (see figure 1.b) and use it for integration in two last levels.

Semantic Web Service technologies enable more flexible interoperability between partners by providing a rich formal language that requester and provider can negotiate by it. Multiple standardization efforts aim to define a framework and a language stack for semantic Web Services, such as OWL-S, WSMO, WSDL-S and METEOR-S. We choose to use WSMO as our model because of its (1) explicit support of mediators and (2) ontological separation of service requester and provider roles [12]. In this section we introduce the architecture of our semantic B2B mediator. As you can see in Figure 2, Semantic B2B Mediator has three components: WSMX, adapter and reasoner.

Figure 2. Semantic B2B Mediator Architecture

WSMX is the integration platform which facilitates the integration process between different systems. The integration process is defined by the WSMX execution semantics, i.e. interactions of middleware services including discovery, mediation, invocation, choreography, repository, etc.

Since WSMX internally operates on the semantic level (WSML), adapters provide transformation functionality for every non-WSML message sent to the B2B gateway. Adapters facilitate lifting and lowering operations to transform between XML instances and WSML instances. Furthermore, back-end adapters are necessary to connect the B2B gateway to the back-end applications of the requester.

The reasoner is required to perform query answering operations on the knowledge base, including the collaboration instance data during execution. We use reasoner for operating consistency checking of instances in run-time. The reasoner has to handle WSML. As yet simple reasoner for some WSML variant is developed. The latest version of Web Service Modelling Toolkit (WSMT) has included these reasoners.

To deploy our solution the followings steps need to be taken. The results of each of these steps will be explained in next sections. There are two main phases to integration requester and provider partners: (1) integration setup phase and (2) integration run-time phase. In the setup phase ontologies are extended or defined. Mapping rules of instances to and from ontology is specified. Also, choreography of partners is described. At run-time mapping rules that specified in setup is performed and process mediator executes choreography of partners.

5. Integration Setup Phase

The core aspect of the setup phase is to model the semantics of services and publish their descriptions. In order to create WSMO services, the ontologies must be created (or reused) together with non-functional, functional and interface descriptions of services.

Semantic Web services are described according to WSMO Service and WSMO Goal definitions. Partners have to describe a WSMO Goal when they are requester and a WSMO Service when they are a provider. Note that WSMO Goal and WSMO Service have similar structural definitions but differ in what they represent. The difference is in the use of defined capability and interface – a WSMO Goal describes a capability and an interface requested by the user whereas a WSMO service describes a capability and an interface provided by a service. WSMO Goals enable goal-based service invocation which is the basis for advanced semantic discovery and mediation provided by the WSMX environment.

5.1. Ontologies

Ontologies describe information models used in semantic service descriptions. We assume that all ontologies are not available up-front and must be created by an ontology engineer. The engineer takes the existing standards and systems as a basis and creates respective ontologies. When creating ontologies, the engineer describes the information semantically, i.e. with richer expressivity as opposed to that of the underlying XML schema.

For brevity, we concentrate on presenting the scenario with partners that utilize RosettaNet e-
5.1.1. Translating XML to WSML. RosettaNet PIPs uses many of associated XML Schemas. For example PIP3A4 have 92 schemas with many namespaces. In sequence we implement a program that explores all files and folders from specific path and aggregates all concepts and relations from “.xsd” files. Three main rules use in this program:

- An XML element is mapped to a WSML concept.
- An XML child element is mapped to a WSML attribute of the concept corresponding to the XML parent element. It has an inferring type of the WSML concept corresponding to the XML child element.
- An XML attribute is mapped to a WSML attribute of the concept corresponding to the XML parent element, with a string constraining type.

5.1.2. Modelling constraints. However, the task doesn’t end with only simply translating the XML Schemas to another richer and formal language like WSML. For complete lifting of XML to WSML we have to model all the constraints on the semantics of the business documents. We include constraints which are not expressible in XML Schema and included in a natural language in the RosettaNet message guidelines respective PIP. For example PIP3B2 has one natural language constraint: HazardousMaterial constraint: If the value of hazardousMaterial is equal to ‘true’ then the occurrence of HazardousMaterialDescription MUST be 1.

According to PIP3B2 documentation, maximum cardinality of “HazardousMaterialDescription” is 1. Table 1 shows the definition of this constraint in WSML.

5.1.3. Verification of Modelled Constraints
For testing modelled constraints we must know if axioms defined in formal way cover business constraints. For this reason, we make all instances that are inadmissible respect to natural language constraints. Then we use a reasoner for consistency checking of these instances one by one. We request reasoner that give us all instances of constrained concept. In response reasoner answer us with an error message with content of violation of considered constraint.

Table 1. Hazardous constraint in WSML-Full

<table>
<thead>
<tr>
<th>concept ShippingContainerItem</th>
</tr>
</thead>
<tbody>
<tr>
<td>HazardousMaterial ofType _boolean</td>
</tr>
<tr>
<td>hasHazardousMaterialDesc ofType _string</td>
</tr>
<tr>
<td>nfp</td>
</tr>
<tr>
<td>dc:relation hasValue HazardousConstraint</td>
</tr>
<tr>
<td>endnfp</td>
</tr>
<tr>
<td>axiom HazardousConstraint</td>
</tr>
<tr>
<td>definedBy</td>
</tr>
<tr>
<td>forall ?x( ?x memberOf ShippingContainerItem</td>
</tr>
</tbody>
</table>
|   and ?x[HazardousMaterial hasValue _boolean("true")]
|   implies |
|   ?x[hasHazardousMaterialDesc hasValue ?y]). |

5.2 Defining Mapping Rules

The mapping rules need to be defined for the runtime phase to lift instance messages to WSML ontology and lower it back to the XML level respectively. There are two options to do that. Either to lift the messages from XML Schemas to WSML and then use a data mediation tool such as the one included in the WSMT to perform the mappings on the ontological level or to directly lift the messages to the domain ontology and essentially implement the mediation in the adapter. In our case, we have chosen the latter option and we perform the using XSLT stylesheets.

As the product definitions in all PIPs are similar, these mapping templates can be reused with all the PIPs. With small modification it is easy to create templates for other e-business frameworks as well.

5.3. Process semantic definition

Choreographies model service behavior describing how service functionality can be consumed by a service requester and, orchestration describing how the same functionality is aggregated out of other services. Interfaces in WSMO are described using Abstract State Machines (ASM) defining rules modeling the interactions performed by the service including grounding definitions to underlying WSDL operations.

Table 2 show a fragment of possible choreography of PIP3A4 that is used for purchase order request.

The choreography is described from the service point of view thus the rule says that in order to send PurchaseOrderConfirmation message, the PurchaseOrderRequest message must be available. By
executing the action of the rule (add(...)), the underlying operation is invoked according to the grounding definition of the PurchaseOrderConfirmation concept which in turn results in receiving instance data from the Web service.

<table>
<thead>
<tr>
<th>Table 2: Ordering Choreography</th>
</tr>
</thead>
<tbody>
<tr>
<td>choreography OrderingChoreography</td>
</tr>
<tr>
<td>stateSignature</td>
</tr>
<tr>
<td>in po#PurchaseOrderRequest</td>
</tr>
<tr>
<td>out po#PurchaseOrderConfirmation</td>
</tr>
<tr>
<td>transitionRules FOChoreographyRules</td>
</tr>
<tr>
<td>forall {?request} with (</td>
</tr>
<tr>
<td>?request memberOf po#PurchaseOrderRequest)</td>
</tr>
<tr>
<td>do</td>
</tr>
<tr>
<td>add( _# memberOf po#PurchaseOrderConfirmation)</td>
</tr>
<tr>
<td>endForall</td>
</tr>
</tbody>
</table>

6. Integration runtime phase

After the set-up phase is completed, semantic B2B mediator is ready for running the processes. We describe here the whole execution process and interactions for a scenario with one requester and many providers:

Adapting. Requester sends out a request in its proprietary format to the back-end adapter. The adapter translates this to WSML and converts it to a goal in WSML and sends it to WSMX.

Discovery. All services in the repository matching the request are found by WSMX discovery. During the discovery, data mediation rules could be executed to resolve differences in the ontologies used for the goal and the service descriptions. After discovery, both the requester’s and the provider’s choreography is registered with the Choreography. Both choreographies are set to a state where they wait for incoming messages that could fire a transition rule. This completes the conversation setup.

Engagement. As discovery operates on abstract description of services, the next step is to find out whether each discovered service can deliver the required product within the given time and give a price for that. In our scenario, engagement is performed by sending request for quote documents and the partners answer those with response documents. Process Mediator first updates the memory of the requester’s choreography with the information that the Purchase Order request has been sent. The Process Mediator then evaluates how data should be added to the memory of the provider’s choreography – data must be first mediated to the ontology used by the provider. Data mediation is performed by execution of mapping rules between both ontologies. Once mediation is complete, the mediated data is added to the provider’s choreography.

Selection and Invocation. Based on the information provided from engagement, the best service is selected. To do this a conversion of different currencies used in quotes is done in WSMX by invoking an appropriate currency transformation service. Then the purchase order process starts with the selective partner.

Returning answer. After the invocation provider returns the purchase order response. After lifting and parsing of the message, the Process Mediator first invokes the mediation of the data to the requester’s ontology and then adds the data to the memory of the requester’s choreography. The next rule of the requester’s choreography can be then evaluated saying that purchase order confirmation message needs to be sent to the provider system. After the message is sent, no additional rules can be evaluated from the requester’s choreography, thus the choreography gets to the end of conversation state. Since both requester’s and provider’s choreography are in the state of end of conversation, the Choreography Engine notifies the execution semantics and the conversation is closed.

7. Expected benefits of solution

We have used specific RosettaNet PIP for our solution setup. As the structure of PIPs is very similar, the results presented in this paper are applicable to the entire RosettaNet framework. It is a time-consuming engineering task to encompass all messages in ontology, but it is a one time effort.

As WSML is a more expressive language than the schema languages used currently, the lifting of PIPs to ontologies can represent more information. Some constraints of PIPs are implicit. By using a formal language we can such as WSML we can express these constraint in a formal and with no ambiguity. Therefore the integration process of partners becomes very easier and faster.

In practice, several e-business frameworks maybe exist. We can use this architecture for partners that don’t use the same e-business frameworks. We have to only define B2B adapter for any framework. It means that by a little change in method that presented in this paper, we can integrate other XML-based e-business frameworks. By means of this architecture we reduce number of mapping from O(n^2) to O(n). In sequence adding new partner in the marketplace is very simpler.

8. Conclusion and Discussion
The scenario discussed in the paper highlights the problems currently observed in B2B collaborations. This problem leads to decrease the chance of competitive sellers that have not any pre-agreement with buyer. Using semantic web service technologies helps sellers to being able to dynamic integrate to buyers without having to make potentially costly changes to their current integration interfaces.

Our semantic B2B mediator allows partner to tackle heterogeneities in B2B interactions. The solution relies upon a formalized message ontology including axiomatized knowledge and rules to resolve data heterogeneities.

We showed how to model and formalize constraints are expressed in natural language. We use a B2B adapter that for transformation of instances from XML to WSML and vice versa. By using a reasoner consistency checking of instances that are created in run-time is carried out.

We defined adaptive executable choreographies, which allow a more flexible integration of sellers. Partner specific rules can be non-obtrusively added to the choreography, which makes it easy to introduce more competition to the supply chain.

References