Semantic Web Services enabled B2B Integration

Title	Semantic Web Services enabled B2B Integration
Author(s)	Kotinurmi, Paavo;Vitvar, Tomas;Haller, Armin
Publication Date	2006

Semantic Web Services enabled B2B Integration

Paavo Kotinurmi[1,2], Tomas Vitvar[1], Armin Haller[1], Ray Richardson[3], and Aidan Boran[3]

[1] Digital Enterprise Research Institute(DERI), Ireland
firstname.lastname@deri.org
[2] Helsinki University of Technology, Finland
firstname.lastname@tkk.fi
[3] Bell Labs Ireland, Lucent
firstnamelastname@lucent.com

Abstract The use of Semantic Web Service (SWS) technologies have been suggested to enable more dynamic B2B integration of heterogeneous systems and partners. We present our approach to accomplish dynamic B2B integrations based on the WSMX SWS environment. 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. This is illustrated through a purchasing scenario. The benefits of applying SWS technologies include more flexibility in accepting heterogeneity in B2B integrations and easing back-end integrations. This allows for example to introduce more competition into the purchasing process within e-business frameworks.

1 Introduction

To integrate heterogeneous enterprise information systems, such as Enterprise Resource Planning (ERP), several e-business frameworks have been developed [10]. The e-business frameworks address B2B integration on business process, message and communication levels [5,6]. As e-business frameworks define B2B integration interfaces between the partners, they can change their internal processes and information systems without a need to change the B2B integration interface. Electronic Data Interchange (EDI) standards, such as EDI X12[4], have been around since the 1970's and are still widely used for B2B integrations. RosettaNet[5] is a widely used XML-based e-business framework.

Companies have invested considerable amounts of money and resources to implement the B2B integrations based on these e-business frameworks and they have the supporting infrastructure largely in place. Back-end integration requires ensuring that internal systems can produce and consume the pre-agreed messages in the collaborative processes. Due to the flexibility of these e-business frameworks regarding message details and message ordering, considerable effort is required to ensure that the B2B integration details of two partners match

[4] http://www.x12.org/
[5] http://www.rosettanet.org/

[13]. Therefore, B2B integrations suffer from long setup times and high costs [7]. This leads to a business models with simple processes, in which long term rigid partnerships are established between organisations. There is, for example, no competition for getting multiple quotes as the default partner is always selected directly for purchasing. This is because there is too much overhead to manage multiple partner specific quoting and purchasing integrations from back-end applications directly.

Semantic technologies and Semantic Web Services (SWS) have been proposed to achieve more dynamic partnerships [1]. The SWS approach based on, for example, OWL-S [4] or the Web Service Modeling Ontology (WSMO) [8] enables annotation of the B2B integration interfaces with semantic information. This allows automated or semi-automated mediation. In addition, SWS enables powerful discovery, composition, and selection capabilities of services. This paper presents a scenario for B2B integration, where a buyer organisation's internal use of SWS technologies enables it to integrate with heterogeneous suppliers that support different e-business frameworks.

The scenario assumes that SWS technologies are introduced to B2B integration stepwise rather than all at once. Instead of designing scenarios of partners using only SWS in the communication, we combine the security, reliability, and scalability strengths of existing e-business frameworks with the SWS's benefits of a more flexible integration with powerful discovery, composition, and selection capabilities.

The SWS solution in this paper is based on the Web Service Modelling eXecution environment (WSMX) [3]. WSMX is a reference implementation of the WSMO and operates using the Web Service Modeling Language (WSML).

The rest of this paper is structured as follows: Section 2 describes a scenario of B2B interactions. Section 3 presents how SWS technologies address the requirements identified in the scenario. Section 4 discusses expected benefits of SWS technologies. Section 5 presents related work and section 6 concludes our solution and discusses topics for further research.

2 Use Case Scenario

We consider an *organisation A* that manufactures electronic devices. For a particular device, organisation A needs specific components that can be delivered by approved suppliers, referred here as *partners B* and *C*. Organisation A needs *X-type of display unit* components that can be delivered by partners B and C. In the current situation, the B2B integration only covers purchasing activities as shown in figure 1 and there is no competition for purchasing per delivery basis. In this proposed scenario, organisation A first submits *Requests For Quotes* (RFQ) to all its suppliers for the components. After the responses, it selects the best quote and initiates the *Purchase Order* (PO) process with the selected partner.

Considering the integrations, the following heterogeneities exist with partners according to general B2B integration levels [5,6]:

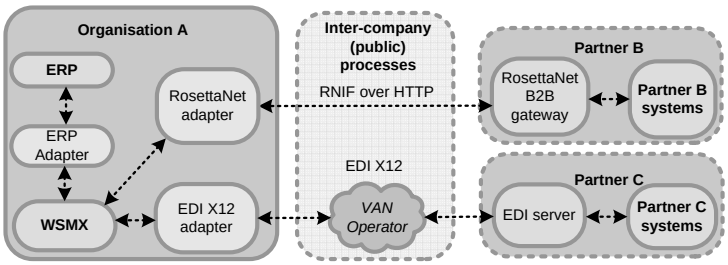

Figure 1. Overall scenario

Communication level interoperation is needed to understand different languages used to describe the messages exchanged and how the message exchange happens. *Partner B* uses the RosettaNet Implementation Framework (RNIF) 2.0 over HTTP(S) for secure communication and the message contents are in XML. RNIF guides how the messages are sent and acknowledged and how digital signatures are used. With *partner C* the communication is achieved via a Value Added Network (VAN) operator, which takes care of the communication between the partners. EDI X12 format messages are put to a file system folder, where the VAN operator collects the messages and ensures the secure delivery of the messages to the partners.

Message level interoperation is the ability to understand exchanged messages (sometimes referred as business documents or payload). RosettaNet defines *Partner Interface Processes* (PIPs) that define standard inter-company process choreographies and the related schemas for the XML messages exchanged. *Partner B* uses the PIP *3A1 Request for Quote* and *3A4 Request Purchase order* messages [6] according to the message guidelines provided by RosettaNet. Both PIPs contain request and response messages. *Partner C* uses EDI X12 messages and expects the *840 Request for Quotation* for quotes and *850 Purchase Order* for orders [7]. The quotes are responded with *879 Price Information* message and the purchase orders by *855 Purchase Order Acknowledgment* message. These PIP and EDI X12 messages use different terms and identifiers in referring to the same concepts. In addition, for example the product identifiers used differ among the companies.

Business Process level interoperation is the ability of companies to exchange messages in the right sequence and timing. *Partner B* complies with PIP 3A1 and 3A4 standard choreographies. That means the partner's response arrives within 24 hours of sending the requests. For every PIP message sent, there is a receipt acknowledgment for delivery. *Partner C* with EDI X12 has not such fixed response times between different messages as it is not dictated by EDI X12. In this case the partner C has agreed to answer the quotes and purchase

[6] http://www.rosettanet.org/pipdirectory
[7] http://www.disa.org/x12workbook/ts/

orders in the same 24 hours. Hence, the choreography differs since the receipt acknowledgment message is not always used with EDI.

3 B2B Integration with SWS Technologies

This section introduces a WSMX enabled architecture to address the requirements of the scenario. We show how SWS technologies can help to mediate the communication, message and process level heterogeneities. We outline some prerequisites for a SWS enabled solution, describe the integration set-up phase using the WSMX and then describe the run-time behaviour. For brevity, we concentrate on presenting the scenario with *Partner B*, who utilises RosettaNet e-business framework.

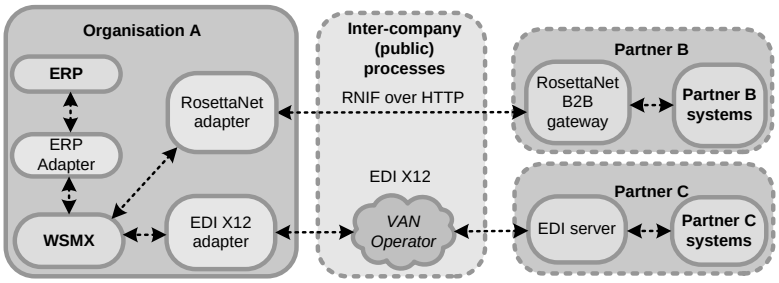

Figure 2. Use Case Scenario with WSMX

3.1 Prerequisites for SWS infrastructure

In this section we analyse the prerequisites organisation A has to address when it sets up the WSMX environment for the B2B integration described above.

Message Ontologies

Based on its requirements, organisation A has to create or ideally reuse *domain ontologies*. In our example these ontologies are used for a formal description of the RFQ and PO process messages. Creating these domain ontologies requires an expert who first understands specific e-business scenarios and second has knowledge about ontology languages to be able to capture information in messages *semantically*. However, since we are still far from an industry wide recognised formal ontology, organisation A in our example needs to define the ontology itself. We assume that organisation A is not in a position to dictate its proprietary ontology to its partners. It is further realistic to assume that it bases its ontology on an existing e-business framework, in our case RosettaNet. This for two reasons, first that organisation A minimises the effort of lifting the RosettaNet PIP XML messages most of its partners still use to the ontological level

and second to further minimise the mapping requirements to its internal ERP system, which still operates on syntactic messages. Figure 3 outlines how the ontology is applied by organisation A and where the lifting/lowering of messages is performed.

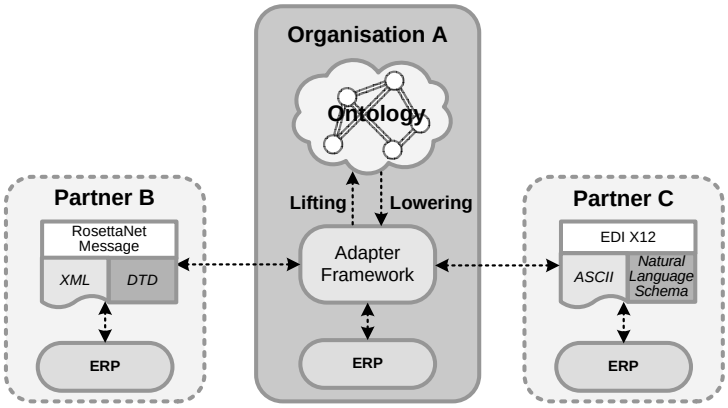

Figure 3. Lifting/Lowering to/from Domain Ontology

The ontology in our scenario ultimately represents simply a different serialisation of the information in the RosettaNet framework with the advantages that it explicitly states logical relationships between elements which can not be expressed in the RosettaNet PIP XML Schemas and DTDs. In RosettaNet this information is included in a natural language document explaining the respective PIP. One of these advantages of the higher expressivity in the WSML ontology language is depicted in the following example.

The DTD versions of PIP 3A1 and PIP 3A4 support two different kind of product identifiers; the Global Trade Identification Number (GTIN), which is recommended by RosettaNet, and company-specific identifiers. The extract in listing 1.1 shows the definition of product identifiers in the PIP 3A1 (and 3A4). The PIP3A1 DTD is very long so only the relevant lines (291-304) are shown.

```
291  <!ELEMENT ProductIdentification
292    (GlobalProductIdentifier?,
293     PartnerProductIdentification*)>
294
295  <!ELEMENT GlobalProductIdentifier
296    (#PCDATA)>
297
298  <!ELEMENT PartnerProductIdentification
299    (GlobalPartnerClassificationCode,
300     ProprietaryProductIdentifier,
301     revisionIdentifier?)>
302
303  <!ELEMENT ProprietaryProductIdentifier
304    (#PCDATA)>
```

Listing 1.1. PIP 3A1 DTD extract

RosettaNet message guidelines for PIP 3A1 add a *natural language constraint* for ProductIdentification that the DTD's expressive power does not capture: *Constraint: One instance of either "GlobalProductIdentifier" or "PartnerProductIdentification" is mandatory.* Without this constraint, a valid ProductIdentification could be without any identifiers as both identifications are optional.

Some of the RosettaNet PIPs have also an XML Schema definitions that can present cardinality constraints for the elements. Listing 1.2 shows an extract of the PIP 3A4 XML schema, where namespaces and annotations are dropped for brevity. The XML Schema has different element names than the ones in DTDs. It also allows arbitrary authorities to specify the identification schemes, which introduces another mapping challenge.

```
<xs:element name="ProductIdentification" type="ProductIdentificationType" />
<xs:complexType name="ProductIdentificationType">
 <xs:complexContent><xs:sequence>
   <xs:element name="ProductName" type="xs:string" minOccurs="0" />
   <xs:element name="Revision" type="xs:string" minOccurs="0" />
   <xs:choice><xs:element ref="AlternativeIdentifier" maxOccurs="unbounded" />
   <xs:element ref="GTIN" /></xs:choice>
 </xs:sequence></xs:complexContent>
</xs:complexType>
<xs:element name="AlternativeIdentifier" type="AlternativeIdentifierType" />
<xs:complexType name="AlternativeIdentifierType">
 <xs:sequence><xs:element name="Authority" type="xs:string" />
 <xs:element name="Identifier" type="xs:string" /></xs:sequence>
</xs:complexType>
```

Listing 1.2. PIP 3A4 XML Schema extract

The product identifier information in the WSML domain ontology is presented in listing 1.3. In this, the GTIN is handled as any other identification authority/qualifier (*qualificationAgency*) and the RosettaNet DTD, XML Schema, and EDI X12 product identification information can be presented in this ontology including the natural language constraints. The qualification agency can be for example the *buyer's, seller's or manufacturer's identifier* or any other identification scheme provider. The axiom in listing 1.3 makes sure that the value of *qualificationAgency* is among those supported for organisation A. Thus, the benefit from applying a more expressive language such as WSML is that it allows the description of logical relationships between the elements. This information can subsequently be applied for better validation of the message contents.

```
244  concept productIdentification
245       nonFunctionalProperties
246            dc#description hasValue "Collection of business properties describing identifiers."
247       endNonFunctionalProperties
248       productIdentifier ofType (1 1) _string
249       qualificationAgency ofType (1 1) _string
250       revision ofType (0 1) _string
251  axiom qualificationAgencyConstraint
252       nonFunctionalProperties
253            dc#description hasValue "The valid list of agencies who have defined product identifiers."
254       endNonFunctionalProperties
255       definedBy !- ?x[qualificationAgency hasValue ?type]
256            and (?type = "GTIN" or ?type = "Manufacturer" or ?type = "Buyer"
257                 or ?type = "Seller" or ?type = "EN" or ?type = "BP").
```

Listing 1.3. Product ontology extract in WSML

Adapters to the Back-end applications

For the integration of back-end systems, the messages used within the ERP system have to be mapped to/from the domain ontology. The ERP adapter is required to perform the lifting/lowering of the internally used messages in the ERP system (e.g. Intermediate Documents (IDocs) in the case of SAP) to the logical framework, i.e. WSML, required by WSMX.

Registration

In current e-business frameworks, a prior agreement between business partners determines with whom the partners do business. For RosettaNet, this includes the specification on the set of PIPs used by the partner in the communication and the *role* for the partner in a certain PIP (e.g. *seller* or *buyer*). In addition, the partners need to provide the *endpoint* information of the *IP address* and the *port*, as well as the public *certificate* used by the partner to sign the messages.

In a SWS enabled integration process a *registration interface* allows a partner to register this information to the SWS environment of organisation A. The registration interface can be accessed by a partner through a web portal or an API of organisation A. By invoking this registration interface, a service description based on a set of PIPs and roles is created and described in WSML. The semantic service description provides all information including the endpoint information necessary to invoke the service.

Adapter framework to external partners

The adapter framework is required to provide a communication interface to partners, who are not able to directly provide WSML compliant messages to WSMX. The adapter framework receives every non-WSML message and acts for WSMX as the actual service. Thus, essentially the adapter functionality is registered as a service with the system. Further, WSMX only operates on the choreography of the adapter (c.f. left part of figure 4), which maps between the choreography of the partner's (c.f. right part of figure 4) e-business environment and the choreography registered with WSMX. The choreography definition is part of the WSMO description of a service and specifies the input and output operations as well as transition rules and constraints on the states in the communication.

Figure 4 shows the RosettaNet adapter execution flow in a PIP process of RFQ request and response messages:

- WSMX first sends the WSML message to the RosettaNet adapter as a WSML RFQ message.
- The adapter receives the RFQ and translates it to a RosettaNet RFQ XML message.
- The adapter creates a RNIF envelope for this message and signs the message using certificates and sends it to the endpoint of partner B (certificate as well

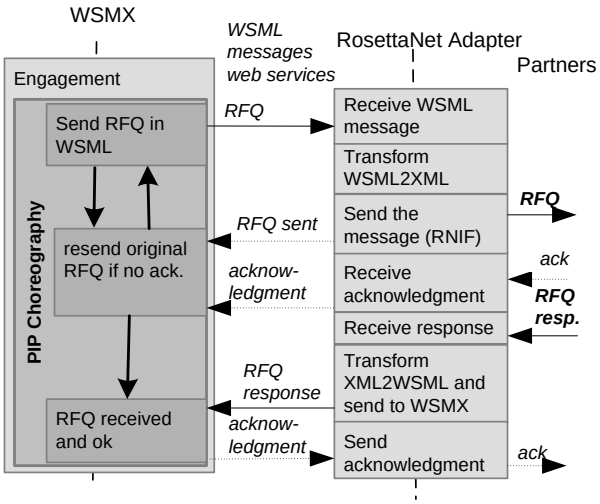

Figure 4. RosettaNet Adapter

as endpoint are implemented in the adapter). As a result, a confirmation that the message has been received is sent back to WSMX.
- WSMX subsequently expects an acknowledgment message by partner B as an RNIF 2.0 signal message.
- After receiving the acknowledgment, WSMX waits for the RFQ response from partner B.
- The adapter receives the RFQ response and translates it using an XSLT script to WSML and sends it to WSMX to check that the response does not violate the axioms of the ontology. This response is processed in WSMX and sent to the back-end applications. WSMX also forwards an acknowledgment signal indicating that their RFQ response was received to the adapter which itself translates it to the message schema expected by the partners.

3.2 Integration Set-up Phase

In the integration set-up phase, the B2B integration for a specific partner is built. The integration set-up phase also includes the registration of the partner's service description with the SWS system of organisation A. Hence, in case a new partner that uses RosettaNet wants to register a service, he needs to provide his endpoint information and choreography details. Partners need to provide this information using the *Registration Interface* described in the previous section. In addition, the information what components they supply is required for discovery.

In case that the new partner uses some other standard or a proprietary format for his messages appropriate adapters and mapping rules might need to be defined.

Creating the adapter to the partners e-business frameworks

The integration of the organisation A and the e-business frameworks of each partner needs specific *RosettaNet* and *EDI X12 adapters*. The role of the adapters is to translate the RosettaNet XML and EDI X12 data formats to WSML and taking care of e.g. RosettaNet-specific RNIF protocol details. For example, the E-business framework message translation based on the mapping rules happens in the WSMX adapter.

- The **communication interface with a partner** is used to send and receive e-business framework-specific messages. It acts as a wrapper for RosettaNet communication with the partner.
  Other basic functionality of the RosettaNet adapter involves the functionality related to enveloping, encrypting and decrypting and validation of RosettaNet messages. Here, the existing B2B gateway functionality can be used if the organisation already has a product for RNIF communication. The RosettaNet adapter needs to have roughly similar functionality to the system presented in [12] with the additional step of XML2WSML and WSML2XML translations. Similarly the ERP and EDI X12 adapters need analogical functionality.
- **Creating data mapping rules from RosettaNet messages to domain ontologies**
  The mapping rules need to be defined for the run-time phase to lift RosettaNet instance messages to the ontology applied by organisation A and lower it back to the XML level respectively. In the scenario mapping rules for PIPs 3A1 and 3A4 are required. 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 Web Services Modeling Toolkit [8] 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. Listing 1.4 contains such an example mapping from a PIP DTD to WSML. Listing 1.5 does the same for XML Schema version PIPs. The mapping lifts the GTIN number to the uniform identification scheme in the ontology. Similarly with EDI X12 the information is lifted to our domain ontology. In the lowering of messages, by knowing that a GTIN identifier and company-specific identifiers point to the same product, the mapping can provide an identifier needed by the given partner. The mapping rules need to be registered in the WSMX ontology repository for run-time mappings. As the product information definitions in all DTD and XML Schema based 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.

[8] http://sourceforge.net/projects/wsmt

```
<xsl:for-each select="ProductIdentification/GlobalPartnerClassificationCode">
  instance localUID memberOf productIdentification
    productIdentifier hasValue <xsl:value-of select="."/>
    qualificationAgency hasValue GTIN
</xsl:for-each>

<xsl:for-each select="ProductIdentification/PartnerProductIdentification/">
  instance localUID memberOf productIdentification
    <xsl:for-each select="ProprietaryProductIdentifier">
      productIdentifier hasValue <xsl:value-of select="."/>
    </xsl:for-each>
    <xsl:for-each select="GlobalPartnerClassificationCode">
      qualificationAgency hasValue <xsl:value-of select="."/>
    </xsl:for-each>
</xsl:for-each>
```

Listing 1.4. DTD-based PIP instance mapping extract

```
<xsl:for-each select="ProductIdentification/GTIN">
  instance localUID memberOf productIdentification
    productIdentifier hasValue <xsl:value-of select="."/>
    qualificationAgency hasValue GTIN
</xsl:for-each>

<xsl:for-each select="ProductIdentification/AlternativeIdentifier/">
  instance localUID memberOf productIdentification
    <xsl:for-each select="Identifier">
      productIdentifier hasValue <xsl:value-of select="."/>
    </xsl:for-each>
    <xsl:for-each select="Authority">
      qualificationAgency hasValue <xsl:value-of select="."/>
    </xsl:for-each>
</xsl:for-each>
```

Listing 1.5. XML-Schema-based PIP instance mapping extract

3.3 Integration Run-time Phase

After the set-up phase is completed, WSMX is ready for running the processes. We describe here the whole execution process and interactions in WSMX according to the scenario: (1) *Converting back-end message to a WSMX goal*, (2) *Discovery* of the possible suppliers capable of fulfilling this request, (3) *Engagement* to negotiate and contract with the discovered suppliers to get the price and condition information, (4) *Selection* of the best supplier, (5) *Invocation* of the PO process with the selected supplier and finally (6) *Returning the answer* to the ERP. The sequence diagram for the run-time behaviour is depicted in figure 5.

- **Converting back-end message to WSMX goal**. Organisation A's ERP system sends out a request in its proprietary format to the back-end adapter. The request is *to get 10 display units X delivered to the plant in Galway, Ireland within 8 days*. The adapter translates this to WSML and converts it to a *goal* in WSML and sends it to WSMX.
- **Discovery**. The execution process starts by invoking the WSMX discovery component. All services in the repository matching the request are found.

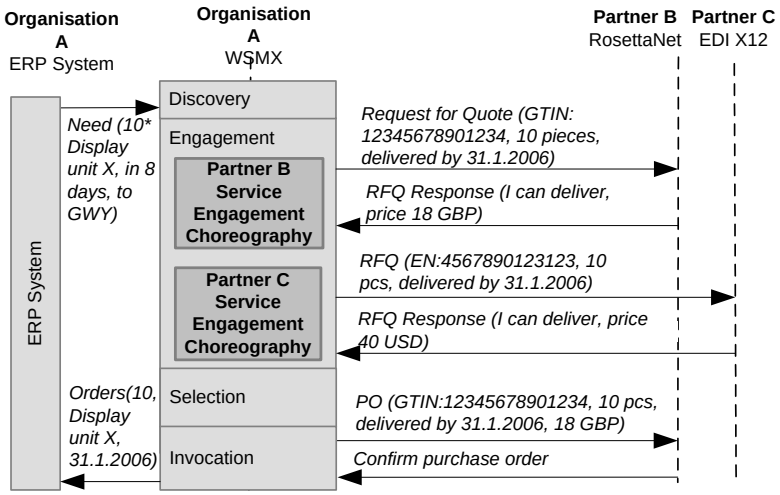

Figure 5. WSMX Process interactions

In our case the services of partners B and C are discovered as potential suppliers. During the discovery, data mediation rules could be executed to resolve differences in the ontologies used for the goal and the service descriptions. However, in our example we only deal with one ontology and all the mediation is done in the lifting and lowering of the XML messages.

- **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 example, engagement is performed for partners B and C by sending RFQ documents and the partners answer those with the RFQ responses. Data and process mapping rules are implemented in the adapters, which handle differences between the RosettaNet and the X12 message exchange patterns (choreographies). Responses coming in RosettaNet PIP 3A1 and EDI X12 879 messages are translated to WSML and sent to WSMX.
- **Selection**. Based on the information provided from engagement, the best service is selected. In this scenario, this is done simply according to the cheapest price. To do this a conversion of different currencies used in quotes is done in WSMX by invoking an appropriate currency transformation service. In our scenario, the partner B has a cheaper quote and is selected.
- **Invocation**. The PO process starts with the partner B using PIP3A4. The concrete interactions between WSMX and partner B happens analogically to the case of the engagement choreography, just the messages exchanged are different.
- **Returning answer**. After the invocation returns the PO response, the necessary data mediation for the product identifiers and currencies expected

by the organisation A's ERP is done. Then the result is sent back to ERP adapter as expected by the ERP system.

4 Expected Business Benefits and Related Work

This section first discusses the scenario and the expected benefits of applying SWS to e-business frameworks and then we discuss WSMX and RosettaNet specific issues.

4.1 Scenario and expected benefits

Our scenario discussed throughout the paper is still simplified and considered only two suppliers. In practice, the number of companies and e.g. proprietary data formats for integration can be larger and that means more mappings to the ontology. However, still the number of required mappings is reduced from n^2 to $2n$. Further the valid suppliers for the given part would need to be checked against the approved manufacturer lists in the product data management systems. In addition, the selection might involve more variables than price, such as history data on supplier quality concerning defects and delivery accuracy. Furthermore, the required service might be something that needs to be configured from a set of services. For example the product manufacturing and transportation could be separated and the different valid combinations should be discovered in a service composition.

As WSML is a more expressive language than the schema languages used currently, the lifting of PIPs to ontologies can represent more information. As a simple example, we provided the mapping of product information to ontologies that captured the natural language constraints and made the "GlobalProductIdentifier" RosettaNet meaning of GTIN number more explicit. The use of formal ontologies enables using common conversion functions to mediate some differences with logical dependencies. RosettaNet currently defines more than 300 *GlobalProductUnitOfMeasureCode*s as a list without any relations to each other. With help of logical relationships, automatic transformations between e.g. "*25 Kilogram Bulk Bag*" and "*50 Pound Bag*" can be done. Currently matching all the details related to PIP messages takes time and only small differences can cause additional system development and testing work. SWS techniques can be applied to describe how companies use the PIPs and automate message compatibility matching, thus making the B2B integration process faster. The resulting integration is more flexible to slightly varying use of messages. For example, different measurement units can be supported easily if they are specified. Adding new partners should be a lot quicker, as a new partner using a domain ontology in describing his services would only need to register the needed details to WSMX before participating in the RFQ processes. Furthermore all this does not require changes to the ERP interfaces. As a result, organisation A would get more quotes to select from.

4.2 WSMX and RosettaNet specific issues

This work utilises existing/planned WSMX components such as data and process mediation. Our contribution is the concept to integrate WSMX with RosettaNet and pointing out how SWS can help to support RosettaNet and EDI X12. As security of communication is not tackled by the current version of WSMX, the use of RNIF for secure communication addresses the security aspects.

There are many challenges related to ontologising RosettaNet specifications. RosettaNet evolves over time. The introduction of XML Schema PIPs have brought some major changes to RosettaNet specifications. The element naming has changed and supporting the same PIP in DTD and XML Schema formats requires own mappings. This involves additional work in defining the necessary mapping rules for different message versions. Having RosettaNet specifications in current non-ontology language means that developers currently need to do this lifting to ontologies. We hope that RosettaNet will adopt an ontology language to formally specify PIP messages as others have also suggested [14].

Our approach requires a lot of work in setting up the SWS infrastructure. However, developing adapters is needed for every e-business framework. The domain ontologies and the mappings are needed with different messages used. After ontologising one PIP process, adding support for other PIPs is easy and exploits in the reuse of mapping rules already one of the benefits of applying an ontology. Furthermore, the RosettaNet adapter behaviour concerning RNIF is identical in all RosettaNet PIPs and thus needs to be defined just once. So far, we have defined the conversations in valid RosettaNet messages and created the domain ontology representing the concepts in the example scenario. The current ontology contains the information carried in those messages rather than all concepts in the RosettaNet PIPs.

5 Related Work

Preist, Trastour, et al. have presented multiple papers on SWS and B2B integration [7,14,13] as research toward similar benefits for B2B integration using semantic web languages. *Preist et al.* [7] presented a scenario and an agent architecture of B2B integration with SWS technologies. In their approach, the partner discovery used service descriptions in the Web Ontology Language (OWL). They also provided a concept of lowering ontology messages and presented ideas of mediating between EDI (EDIFACT) and RosettaNet messages. *Trastour et al.* [14] augment RosettaNet PIPs with partner-specific DAML+OIL constraints to overcome the shortfalls of RosettaNet. They want to determine if parties have compatible processes and messages, and automatically propose modifications if not. *Trastour et al.* [13] use agent communication to help in negotiation and contract forming (engagement) processes for making B2B integration faster. Common to all these three papers is that they expect the participating companies to at least partly use semantic web languages in the inter-company communication. Furthermore, neither the non-functional security properties nor integration to back-end systems are addressed in the papers. Our work goes a

step further in defining details how the B2B integration with RosettaNet using a SWS environment can be achieved and how to integrate existing back-end systems with it.

Many papers present B2B integration solutions that do not use any SWS technologies, but still have more similarities to our work. *Dogaz et al.* [2] present an implementation where an ebXML infrastructure is developed by exploiting the UDDI registries and RosettaNet PIPs. The UDDI registry is used to store ebXML documents and process descriptions that correspond to WSMX registries described here. They also provide a solution for secure communication. *Sundaram and Shim* [11] present an infrastructure for B2B exchanges with RosettaNet. They have a three-tier client-server prototype that allows customers to send PIP messages using a browser. *Sayal et al.* [9] present a tool, that supports RosettaNet PIPs and allows generating complete processes from PIPs by taking internal integration needs into account. These approaches are more static and lack both the mediation capabilities enabled by SWS and secure communication. Common to all these related works is a lack of the concept for integrating with existing back-end systems.

6 Conclusions and Future Work

We presented a scenario and a supporting WSMX Semantic Web Service (SWS) architecture for B2B integration, where a buyer organisation communicates with partners using RosettaNet and EDI X12. We showed how the interoperation aspects are handled from the buyer organisation's point of view and how the use of SWS technologies enables communication with its heterogeneous partners. We demonstrated parts of the ontologising process of existing messages and how they can be used to mediate the differences in B2B integrations. We particularly showed how the product identification information can be captured in WSML ontology language covering current RosettaNet and EDI X12 definitions. We also presented the functionality of a RosettaNet adapter that is needed to adapt the WSML ontology language and web services used in WSMX to RosettaNet communication requirements.

As future work, we plan to show more benefits of using formal ontologies for B2B integration and extend the examples to include more complex product data and axioms related to valid product individuals. This scenario had rather static choreography descriptions, but we plan to extend this to more complex choreographies and to apply process mediation rules to non-matching choreography descriptions.

Acknowledgments

This material is based upon works supported by the Science Foundation Ireland under Grant No. SFI/02/CE1/I131 and the EU funded Knowledge Web project (FP6 - 507482). This work is also partly supported by the Finnish Funding Agency for Technology and Innovation (Tekes) and the Graduate School for Electronic Business and Software Industry.

References

1. C. Bussler, D. Fensel, and A. Maedche. A Conceptual Architecture for Semantic Web Enabled Web Services. *SIGMOD Record*, 31(4):24 – 29, 2002.
2. A. Dogac, Y. Tambag, P. Pembecioglu, S. Pektas, G. Laleci, G. Kurt, S. Toprak, and Y. Kabak. An ebXML infrastructure implementation through UDDI registries and RosettaNet PIPs. In *Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data*, pages 512–523, 2002.
3. A. Haller, E. Cimpian, A. Mocan, E. Oren, and C. Bussler. WSMX – A Semantic Service-Oriented Architecture. In *Proceedings of the 3rd International Conference on Web Services*, pages 321 – 328, Orlando, Florida, USA, 2005. IEEE Computer Society.
4. D. Martin et al. Owl-s: Semantic markup for web services. Member submission, W3C, 2004. Available from: http://www.w3.org/Submission/OWL-S/.
5. B. Medjahed, B. Benatallah, A. Bouguettaya, A. H. H. Ngu, and A. K. Elmagarmid. Business-to-business interactions: issues and enabling technologies. *VLDB Journal*, 12(1):59–85, 2003.
6. J.-M. Nurmilaakso and P. Kotinurmi. A Review of XML-based Supply-Chain Integration. *Production Planning and Control*, 15(6):608–621, 2004.
7. C. Preist, J. E. Cuadrado, S. Battle, S. Williams, and S. Grimm. Automated Business-to-Business Integration of a Logistics Supply Chain using Semantic Web Services Technology. In *ISWC '05: Proceedings of 4th International Semantic Web Conference*, 2005.
8. D. Roman, U. Keller, H. Lausen, J. de Bruijn, R. Lara, M. Stollberg, A. Polleres, C. Feier, C. Bussler, and D. Fensel. Web Service Modeling Ontology. *Applied Ontologies*, 1(1):77 – 106, 2005.
9. M. Sayal, F. Casati, U. Dayal, and M.-C. Shan. Integrating Workflow Management Systems with Business-to-Business Interaction Standard. In *Proceedings of the 18th International Conference on Data Engineering*.
10. S. S. Y. Shim, V. S. Pendyala, M. Sundaram, and J. Z. Gao. Business-to-Business E-Commerce Frameworks. *IEEE Computer*, 33(10):40–47, 2000.
11. M. Sundaram and S. S. Y. Shim. Infrastructure for B2B Exchanges with RosettaNet. In *Proceedings of the Third International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems*, pages 110–119, 2001.
12. J. Tikkala, P. Kotinurmi, and T. Soininen. Implementing a RosettaNet Business-to-Business Integration Platform Using J2EE and Web Services. In *7th IEEE International Conference on E-Commerce Technology*, pages 553–558. IEEE Computer Society, 2005.
13. D. Trastour, C. Bartolini, and C. Preist. Semantic Web support for the business-to-business e-commerce pre-contractual lifecycle. *Computer Networks*, 42(5):661–673, 2003.
14. D. Trastour, C. Preist, and D. Coleman. Using Semantic Web Technology to Enhance Current Business-to-Business Integration Approaches. In *Proceedings of the 7th International Enterprise Distributed Object Computing Conference*, pages 222–231. IEEE Computer Society, 2003.