Handling heterogeneity in RosettaNet messages

Handling heterogeneity in RosettaNet messages

Armin Haller
DERI
National University of Ireland, Galway
armin.haller@deri.org

Paavo Kotinurmi
Helsinki University of Technology
Finland
paavo.kotinurmi@tkk.fi

Tomas Vitvar, Eyal Oren
DERI
National University of Ireland, Galway
firstname.lastname@deri.org

ABSTRACT

We present a semantic B2B gateway based on the WSMX semantic Service-Oriented Architecture to tackle heterogeneities in RosettaNet messages. We develop a rich RosettaNet ontology and use the axiomatised knowledge and rules to resolve data heterogeneities and to unify unit conversions. We use adaptive executable choreography definitions to easily integrate new sellers into existing RosettaNet collaborations.

Categories and Subject Descriptors

D.2.12 [**Software**]: Software Engineering; Interoperability[Data mapping]

Keywords

B2B collaboration, RosettaNet ontologies, ontology mapping, partner integration methodology, adapter framework

1. INTRODUCTION

Traditional B2B integrations suffer from long implementation times and high costs [7], leading to long-term rigid partnerships. RosettaNet[1] is a prominent standard for B2B integration that represents an agreement on the message exchange patterns, the message content and a secure transportation mechanism between companies operating in the IT and electronics industries. The message content of structurally valid RosettaNet Partner Interface Processes (PIP) is defined by either DTD for the older PIPs or XML Schema for the newer ones. However, the interoperability challenges are only partly solved:

- The schema languages lack expressive power to capture all necessary constraints and do not make all document semantics explicit. A feature which is even advertised by RosettaNet experts [3] to be lacking in the current specifications.
- Companies can use the same PIP messages differently as the messages contain many optional elements.
- Companies can support different parts of the RosettaNet dictionaries.

When the number of partners increases such limitations become increasingly important. Since resolving heterogeneities on a case-by-case basis is expensive, buyers often use only one seller. Thus, B2B integration solutions are built in a way that more competitive arrangements are not easily supported.

We propose a semantic B2B gateway, describe the deployment methodology in a RosettaNet collaboration scenario and introduce the necessary ontologies. We apply semantic Web Service technologies to RosettaNet collaborations, but we do not impose the use of these technologies to the business partners. In our solution only the Buyer uses a semantic B2B gateway, the partners keep their current RosettaNet interfaces. Our main contributions are:

- We define an non-obtrusive B2B gateway architecture based on an existing semantic Service-Oriented Architecture solution, its deployment methodology and its implementation. As such, our solution represents a generic B2B adapter for a RosettaNet B2B collaboration.
- We encode the information exchanged in RosettaNet PIP messages in a formal ontology, including (i) constraints that cannot be represented in current PIP message schemes such as cardinality constraints spanning multiple fields, (ii) definitional facts that constrain the interpretations of elements in RosettaNet messages such as conversions between systems of measurement, and (iii) domain-specific rules.
- We show how our solution uses a constraint-based choreography language developed in the Web Service Modeling Ontology (WSMO) [8] to allow an easy integration of new partners to a collaboration.

The paper is structured as follows: first we motivate our solution using an example RosettaNet quoting and purchasing scenario in section 2. We give a brief introduction to semantic Web Services in section 3. Section 4 presents the architecture of our semantic B2B gateway and describes its deployment methodology. Section 5 introduces our ontologies that are used in the deployment. Section 6 discusses the generalisability of our proposed solution. Section 7 positions our work to related literature and we conclude in section 8.

[1]See `http://www.rosettanet.org`.

SAC'07 March 11-15, 2007, Seoul, Korea

2. MOTIVATING EXAMPLE

As we assume gradual introduction of semantic Web Service technology to RosettaNet collaborations, we present a quoting and purchasing scenario which is already implemented according to the RosettaNet guidelines of PIP3A1 and PIP3A4. Figure 1 shows the overall choreography including the message exchange of a Buyer (requester) and a Seller (provider) using BPMN[2] notation. The white coloured activity boxes denote parts of the internal computational steps, dotted boxes are placeholders for possibly many omitted computation steps performed internally, whereas the dark coloured boxes represent the public behaviour according to PIP3A1 and PIP3A4 respectively.

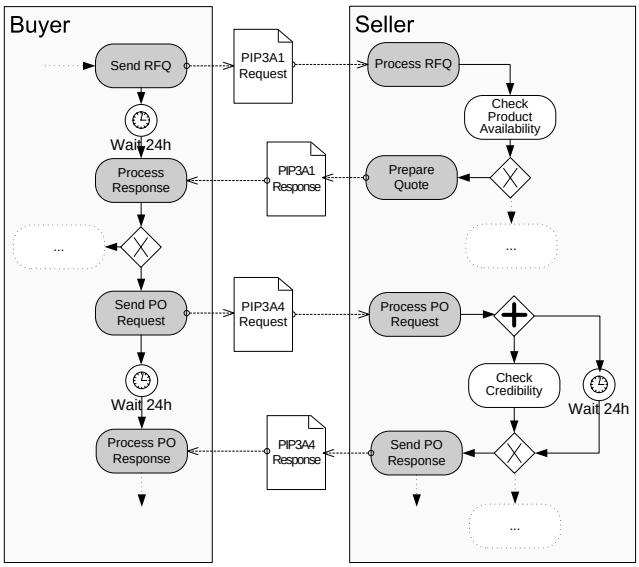

Figure 1: RosettaNet B2B Collaboration

Figure 1 shows the current interaction with only one Seller. However, if the Buyer wants to introduce more competition to get multiple quotes from different sellers, the following challenges arise:

- RosettaNet defines a schema for every message exchanged in a PIP, but many elements within the schema are optional and not implemented by every company. Thus, it might be necessary for the Buyer to interpret additional information sent by a newly introduced Seller. For example, the new Seller might offer substitutable products in its responses.
- The internal processes according to the global choreography have to be changed for the introduction of every new Seller. Not only that multiple messages have to be sent and received, also the decision process to determine which Seller is chosen for purchasing has to be introduced.
- The RosettaNet business dictionary gives a finite set of enumerated values for certain elements in a PIP message. The relations between these values are not defined. This leads to heterogeneity in the messages as for example EU sellers typically use Euro values for quotation and the Metric system for units of measurement, whereas U.S. based companies quote in Dollars and commonly use inch-pound units. The significance of this problem can be seen in the widespread use of such elements in the different RosettaNet messages. We took a sample of 56 PIP message specifications, covering product information, order management, inventory management and manufacturing clusters which represent 29,5% of the total 190 message specifications in RosettaNet. We found that measuring units were used in 24 message types (43%) and currency information was used in 28 message types (50%).

To overcome these challenges we introduce semantic Web Service technology to such RosettaNet collaborations. We encode the information exchanged in the RosettaNet PIP3A1 messages in a formal ontology, including definitional facts that make the dependencies between elements defined in the RosettaNet business dictionary explicit. Further, our solution uses a constraint-based choreography language which allows us to introduce partner-specific behaviour, including different mapping relations for heterogenous message content.

3. SEMANTIC WEB SERVICES

Semantic Web Service technologies enable more flexible interoperability between partners by describing the requirements and offers of service requesters and providers in a rich formal language. Multiple standardisation efforts aim to define a framework and a language stack for semantic Web Services, such as OWL-S, WSMO and METEOR-S[3].

We have chosen the Meta Model offered by the Web Service Modeling Ontology (WSMO) and its accompanying ontology language to model and implement the examples described in our paper. The choice has been made on the fact that there is a semantic Service-Oriented Architecture readily available implementing the technologies introduced in WSMO. Although we have opted for one such framework the solutions presented could theoretically also be implemented using one of the other semantic Web Service frameworks.

WSMO [9] provides a conceptual model and a language for describing the relevant aspects of services, including, but not limited to those accessible as Web Services. The goal of such markup is to enable the automation of tasks (e.g. discovery, selection, composition, mediation, execution and monitoring) involved in both intra- and inter-enterprise integration. WSMO also defines a model to describe the choreography and orchestration of a Web Service. Both are based on a variant of the formalism of Abstract State Machines [8].

The markup of services according to the WSMO conceptual model is expressed in the Web Service Modeling Language (WSML) [2] family of ontology languages. WSML consists of a number of variants based on different logical formalisms which correspond to different levels of logical expressiveness and are both syntactically and semantically layered.

WSMO is the underlying model of the Web Service Execution Environment (WSMX) [5]. WSMX is an integration platform conforming to the principles of a Service-Oriented

[2]See `http://www.bpmn.org/`.

[3]See `http://www.w3.org/2002/ws/swsig/` for specifications of the different standards

Architecture which facilitates the integration between different systems. The integration process is defined by adaptive operational semantics, defining the interactions of middleware services including discovery, mediation, invocation, choreography, repository services, etc. Thus, WSMO, WSML and WSMX provide a coherent framework that covers all aspects of semantic Web Services.

4. SEMANTIC B2B GATEWAY

In this section we introduce the architecture, implementation and deployment methodology of our semantic B2B gateway. This gateway operates on ontologies that capture rich semantic information about the RosettaNet standard, domain-specific constraints and the process model of the business collaboration. Some of these ontology parts are generic (e.g. relations between systems of measurement), others are specific to the business collaboration. Accordingly, in section 5 we first define the generic ontology parts and then give examples for the collaboration-specific parts.

4.1 Architecture

Our semantic B2B gateway is a light-weight adoption of the WSMX architecture [12] and relies on four components (knowledge base, choreography engine, adapter framework and reasoner) as depicted in figure 2.

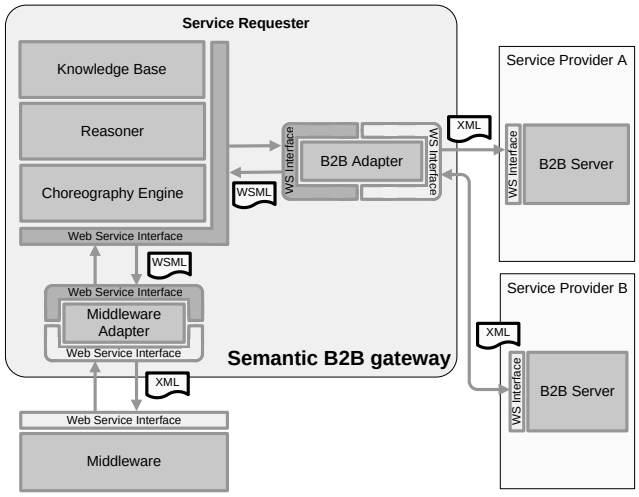

Figure 2: Overview of B2B gateway architecture

Our architecture is based on the WSMO framework. WSMO is mostly targeted towards dynamic discovery of providers, achieved by matching the description of a requester's goal with the description of a provider's service capability. However, our solution omits the WSMO ontology parts concerned with dynamic discovery (goal and capability) of semantic Web Services, but operates on the WSMO service interface only. It is a description of the communication patterns according to which a service requester consumes the functionality of the service. We disregard the functional description mainly because current business practice in RosettaNet collaborations does not consider an integrated discovery and invocation of services. The "discovery" of business partners is conducted when the infrastructure is set up and is commonly based on well-established and long-running business relations.

4.1.1 Knowledge Base

The knowledge base contains the generic and collaboration-specific knowledge required for resolving the heterogeneities in the collaboration. Specifically, these are the RosettaNet ontology, the domain specific rules and the choreography specifications. It is further populated at run-time with ontology instances generated for every incoming RosettaNet message. The knowledge base is implemented by the WSMX resource manager, which coordinates the access to (distributed) repositories.

4.1.2 Choreography Engine

The semantic B2B gateway manages the full life cycle of a RosettaNet PIP. The collaboration described in the PIP is expressed as a WSMO choreography and its execution is managed by the choreography engine (provided by WSMX). The engine sends and receives the exchanged messages and updates the state of the choreography according to the message content. Although choreography languages are commonly used as non-executable descriptions, in our case they are executed to control the message exchange in the system.

WSMO choreographies are modelled as Abstract State Machines and are processed using standard algorithms during runtime. The current state in the execution is represented by ontology instances. According to the instance data, a transition rule is selected from the rule base within a choreography. The rule is interpreted and the ontology instance is modified accordingly. It is the responsibility of the choreography engine to maintain the state of a conversation and to take the correct action when that state is updated. For example, the message received from a service provider updates the state of a choreography instance.

As such the choreography descriptions are used as constraints on the partner interaction. They can therefore easily be changed and extended when new partners are introduced into a collaboration. New transition rules can be added non-obtrusively, which trigger on certain parts of a message sent by one partner only (c.f. section 4.2).

4.1.3 Adapter Framework

The adapter framework provides transformation functionality for every non-WSML message sent to the B2B gateway. Adapters are necessary for lifting and lowering syntactical to semantical representations, namely XML instances in the messages sent from the partners to WSML ontology instances. Furthermore, middleware adapters are necessary to connect the B2B gateway to the back-end applications of the requester.

The adapters act as the actual service provider for the semantic B2B gateway. The service interface of the adapter is used by the gateway to invoke the provider functionality instead of the RosettaNet service endpoint of the partner. Thus, the adapter is responsible for executing the correct endpoint of the partner service. However, adapters only perform data manipulation, their interface behaviour replicates the behaviour of the underlying partner service.

4.1.4 Reasoner

The reasoner is required to perform query answering operations on the knowledge base, including the collaboration instance data during execution. The reasoner has to handle WSML and should have built-in predicates for handling basic data-types, basic arithmetic functions as well as basic

comparison operators. Dedicated reasoners for WSML are under development.

4.2 Deployment Methodology

In this paragraph we outline the deployment methodology of our semantic B2B gateway. To deploy our solution the followings steps need to be taken by the Buyer. The results of each of these steps will be explained in section 5.

1. First, the RosettaNet ontology has to be extended for a specific PIP. Once a complete RosettaNet ontology is defined, this step becomes superfluous.
2. The transformation rules to communicate with the middleware adapter have to be built. This step is similarly necessary in traditional B2B gateways when connecting them to the back-end applications.
3. Next, lifting and lowering rules to and from the ontology have to be defined. In traditional B2B gateways such transformation rules are developed between every two message schemas. Our ontology acts as a unifying model and minimises the transformation effort, since mappings can be reused in multiple schema transformations.
4. The choreography description representing the PIP collaboration process has to be defined for every collaboration. It is used for the execution of the collaboration. In contrast to traditional B2B gateways it abstracts from the implementation of the internal processes and no changes have to be made to such, if the choreography changes.
5. For possible heterogeneities with a specific partner in the collaboration, domain-specific conversion functions have to be included in the knowledge base, such as conversions between units of measurement. They can be reused for every other occurrence of the same mapping problem. In traditional B2B gateways such heterogeneities would be encoded in the mapping script or handled in the back-end applications on a case-by-case basis.
6. For conversion scripts that cannot be represented in the ontology (such as currency conversions that need to consult an external service), the choreography has to be extended with rules that trigger external scripts to resolve such data heterogeneities in partner message.
7. Finally, to avail of the benefits of the homogenised information (including the instance data at run-time) queries have to be written, which populate the data back to the back-end applications.

For each additional partner only choreography rules to resolve specific heterogeneities have to be introduced.

5. ONTOLOGIES

In this section we explain the steps of the deployment methodology in more detail by subsequently discussing (i) the generic RosettaNet ontology that underlies all further descriptions, (ii) the choreography ontology that describes the collaboration process, and (iii) the syntactic transformation rules that are interpreted in the adapters.

5.1 RosettaNet Ontology

A domain ontology to formally capture the message content exchanged in RosettaNet collaborations is required. Ideally, existing domain ontologies should be reused. Since there is not yet an industry wide recognised ontology for business messages we have constructed such an ontology ourselves. Naturally, it is based on the RosettaNet specification which itself can be regarded as a light-weight ontology.

Apart from translating the schema specifications to the richer and formal ontology language WSML, we also need to model the constraints on the semantics of the business documents. Our ontology includes constraints that are not expressible in DTD or XML Schema and that capture implicit knowledge provided in the RosettaNet message guidelines and accompanying documentation.

The ontology is modelled according to PIP3A1, containing concepts such as "PartnerDescription" or "PhysicalAddress", and their attributes. These concepts are straightforwardly expressed in WSML and not discussed here, the ontology can be found at `http://m3pe.org/ontologies/rosettaNet/core.wsml`. Instead, we focus on the modelling of richer constraints, which cannot be expressed with technology used within the RosettaNet specifications.

5.1.1 *Definitional Facts*

An example of such a richer cardinality constraint not expressible in current RosettaNet messages schemas, is the constraint imposed on the "BusinessDescription" element used in all RosettaNet PIPs. The "BusinessDescription" element includes business properties that describe a business identity and its location. At least one of the possible three subelements "businessName", "GlobalBusinessIdentifier" or "PartnerBusinessIdentification" must be provided in order to make a valid PIP. Such constraints can easily be included in our ontology using WSML notation [2] as shown below in listing 1:

```
concept businessDescription
    businessname ofType (0 1) businessName
    globalbusinessidentifier ofType (0 1) globalBusinessIdentifier
    partnerbusinessidentification ofType partnerBusinessIdentification
    nfp
        dc#relation hasValue validBusinessDescription
    endnfp
axiom validBusinessDescription
    definedBy
        forall ?x,?y (?x memberOf businessDescription implies
            ?y memberOf businessName or
            ?y memberOf globalbusinessidentifier or
            ?y memberOf partnerbusinessidentification).
```

Listing 1: Rich cardinality constraints example

Listing 2 shows examples of implicit knowledge captured in our RosettaNet ontology. For example, the RosettaNet dictionary has a list of 335 possible values for units of measurements, with the logical relationships between values unspecified. We made such logical relations explicit and included these axiomatisations in our ontology. The first axiom "resolveMeasurementUnitType" in listing 2 shows how the measurement units defined with natural language text in the RosettaNet PIPs can be resolved to their corresponding numerical value. The second part of the listing defines a function used to relate a kilogram value to its equivalent pound value. As such the Buyer can query the knowledge base and retrieve instances data of different sellers with homogenised values for measurement units.

```
axiom resolveMeasurementUnitType
  definedBy
    forall ?x(?x[globalProductUnitOfMeasurementCode hasValue "
          dozen"] memberOf quoteLineItem implies ?x[
          globalProductUnitOfMeasurementCode hasValue "12"]).
    forall ?y(?y[globalProductUnitOfMeasurementCode hasValue "10-
          pack"] memberOf quoteLineItem implies ?y[
          globalProductUnitOfMeasurementCode hasValue "10"]).

relation poundKilo (ofType productQuantity, ofType productQuantity)
  nfp
    dc#relation hasValue poundKiloDependency
  endnfp

axiom poundKiloDependency
  definedBy
    forall ?x,?y (
      poundKilo(?x,?y) equivalent
        ?x memberOf productQuantity and
        ?x[globalProductUnitOfMeasureCode hasValue "Kilogram"]
          memberOf quoteLineItem and
        ?y memberOf productQuantity and
        ?y[globalProductUnitOfMeasureCode hasValue "Pound"]
          memberOf quoteLineItem and
        ?x = wsml#numericDivide(?y,?x,0.45359237)).
```

Listing 2: Definitional facts example

5.1.2 *Domain-specific Rules*

Each collaboration requires the setup of additional domain-specific rules to capture any data heterogeneity that is not resolved by the definitional facts in the domain ontology.

These domain specific rules (conversion relations in our case) define how attribute values in the different WSML instances can be transformed. One such example is given in listing 3. It defines a function to calculate the unit price by taking the "financialAmount" and "productQuantity" given in the RosettaNet ontology instance. This rule can be used by the requester to compare the prices of two or more partners. The "financialAmount" in a PIP3A1 message can refer to different quantities of the product. The dependencies between the different packaging sizes and its corresponding values are already made explicit in the ontology as described in the previous section. Together with the rule defined in listing 3 they form the basis to query the knowledge base to automatically compare the prices on a per-unit basis.

```
relation unitPrice (ofType financialAmount, ofType productQuantity,
      ofType decimal)
  nfp
    dc#relation hasValue unitPriceDependency
  endnfp

axiom unitPriceDependency
  definedBy
    forall ?x,?y,?z (unitPrice(?x,?y,?z) equivalent
        ?x memberOf financialAmount and
        ?y memberOf productQuantity and
        ?z = wsml#numericDivide(?z,?x,?y)).
```

Listing 3: Domain-specific conversion example

5.2 Choreography Ontology

The choreography ontology defines the interface behaviour of the requester based on the expected input from the middleware and the collaboration partners. To collaborate with its partners, the choreography interface of the requester should comply with the interface behaviour of the partner that provides a quote response. Since all suppliers in our supply chain use RosettaNet, there is already agreement on the message exchange patterns. However, there are still mismatches on the message content that is sent and received in the collaboration. These definitions are specific to a collaboration.

We describe an example of a possible choreography for PIP3A1 in listing 4. For space consideration we only show a snippet of the choreography description[4]. Please note that the "//..." symbol denotes parts omitted in the listing. The namespace declarations are also omitted in the listing.

```
choreography
  stateSignature
    importsOntology {
      _"http://www.wsmx.org/ontologies/rosetta/coreelements",
      _"http://www.m3pe.org/ontologies/rosetta/CTRLASM"
      }
    out rfq#Pip3A1RFQRequest withGrounding _"http://example.org/
        webServices#wsdl.interface(ServicePortType/RFQ/Out)"
    out curr#currConvRequestUsdEur withGrounding _"http://www.
        webcontinuum.net/webservices/ccydemo.wsdl#"
    in rfq#Pip3A1RFQResponse withGrounding _"http://example.org/
        webServices#wsdl.interface(ServicePortType/RFQ/In)"
  transitionRules
    if (?Pip3A1RequestForQuoteRequest[
      fromRole hasValue ?fromRole,
      globalDocumentFunctionCode hasValue
        ?globalDocumentFunctionCode,
      quote hasValue ?quote,
        thisDocumentGenerationDate hasValue
      ?thisDocumentGenerationDate,
        thisDocumentIdentifier hasValue ?thisDocumentIdentifier,
        toRole hasValue ?toRole
    ] memberOf rfq#Pip3A1RFQRequest) and
    //...
      update(?controlledState[currentState hasValue 1] memberOf
          ctrlasm#controlledState)
    endIf

    if (?controlledState[
      currentState hasValue 1
      ] memberOf ctrlasm#controlledState) and
    exists ?Pip3A1RequestForQuoteRespond
      (?Pip3A1RequestForQuoteRespond memberOf rfq#
          Pip3A1RFQResponse) and
    //...
    update(?controlledState[currentState hasValue 2] memberOf
        ctrlasm#controlledState)
  endIf

  if (?controlledState[
    currentState hasValue 2
    ] memberOf ctrlasm#controlledState) and
    exists ?globalProductSubstitutionReasonCode,
        ?productIdentification
      (?substituteProductReference[
          globalProductSubstitutionReasonCode hasValue
        ?globalProductSubstitutionReasonCode,
          productIdentification hasValue ?productIdentification
          ] memberOf core#substituteProductReference) then
          add(_# memberOf rfq#Pip3A1RFQRequest)
    endIf

  if (?controlledState[
    currentState hasValue 2
    ] memberOf ctrlasm#controlledState) and
      exists ?globalCurrencyCode, ?monetaryAmount
      (?totalPrice[
        globalCurrencyCode hasValue ?globalCurrencyCode,
        monetaryAmount hasValue "USD"
      ] memberOf rfq#totalPrice) then
          add(_# memberOf curr#currConvRequestUsdEur)
      endIf
```

Listing 4: Choreography in WSML

[4]See `http://www.m3pe.org/ontologies/rosettaNet/` for the full choreography ontology.

State signature As described in section 4.1.2 a WSMO choreography definition consists of a state signature (line 32–39 in listing 4), which imports one or possibly many ontologies. In our example we import two ontologies, the message ontology capturing concepts in RosettaNet messages and a control State ASM ontology which allows us to define termination and to impose an explicitly modelled control flow for certain parts of the choreography.

Rules for data heterogeneity The transition rules (line 40–382 in listing 4) are basic operations, such as adding, removing and updating the instance data of the signature ontology. The transition rules in listing 4 capture only a small number of heterogeneities possibly occurring in a PIP3A1 collaboration. New rules can be added when new partners are introduced into the scenario with different data heterogeneities.

The first transition rule (line 41–172) defines the ontology schema of the message sent by the Buyer (the mode "out" in the "stateSignature" states the passing direction, thus that the message is sent) and that the "currentState" of the choreography instance is updated to the value "1" after a successful transmission of the message. The grounding, denoted by "Pip3A1RFQRequest" in the state signature as described above, only defines one WSDL interface. However, since the Buyer in our example wants to get quote responses from multiple sellers, the actual grounding list would include one endpoint for every partner.

The second rule fires (line 174–358) when a quote response is received from one seller. The rule acts as a schema validation such that it only triggers if the message returned by a partner includes all required fields. For space considerations these parts of the choreography (line 179–356) are not shown in listing 4.

The last two transition rules are introduced to the model to allow a partner-specific handling of message content. The third transition rule (line 360–371) fires if a partner provides a substitute product element in the PIP3A1 response message. It results in the creation of a new quote request.

The fourth transition rule is introduced to homogenise the amount if the item is quoted in a different currency to the one the requester uses. Since currency exchange rates are subject to constant changes the relations between them can not be made explicit in the ontology. The conversion has to be performed at run-time and the derived facts are added to the knowledge base. As an example of such, the fourth transition rule (line 373–382) ensures that a currency conversion service is used if the amount in the message is quoted in USD.

5.3 Syntactic Transformation Rules

The adapters transform all non-WSML messages sent to the B2B gateway: they lift and lower between syntactical and semantical representations. We can identify two types of adapters: the B2B adapters that map between RosettaNet messages and WSML and the middleware adapters that map between (possibly proprietary) message schemas of the backend applications and WSML.

The B2B adapters operate on transformation rules written as XSLT scripts; parts of such kind of rules have been previously shown by Kotinurmi et al. [6].

6. GENERALISABILITY OF OUR SOLUTIONS

Although we have used one specific RosettaNet PIP, the results presented in this paper are applicable to the entire RosettaNet framework. The ontology we built for PIP3A1 specifically includes common elements for every PIP message. It is a time-consuming engineering task to encompass all messages in RosettaNet in the ontology, but it is a one time effort. Since such a comprehensive ontology is not defined yet, one would extend our ontology on a case-by-case basis.

As we analysed, the axioms we have defined for PIP3A1 to resolve heterogeneities in units of measurement and to make the relations between values in the RosettaNet business dictionary explicit, are applicable to almost half of all RosettaNet messages.

The implicit knowledge we have captured in the ontology could also be resolved with XSLT scripts or custom codings. However, the main advantage of expressing these kind of definitional facts in an ontology is its reusability. Scripts have to be written for every schema conversion causing point-to-point integrations that are costly to develop and to maintain.

7. RELATED WORK

There are a number of papers discussing the use of semantic Web Service to enhance current B2B standards. Some concentrate on ontologising B2B standards [4, 1]. Foxvog and Bussler describe how EDI X12 can be presented using WSML, OWL and CycL ontology languages [4]. The paper focuses on the issues encountered when building a general purpose B2B ontology, but does not capture RosettaNet in particular and describes the ontology engineering methodology only. Anicic et al. [1] present how two XML Schema-based automotive B2B standards are lifted using XSLT to OWL-based ontology. They use a two-phase design and runtime approach similar to our's. The paper 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, 10, 11]. Preist et al. [7] 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 formalised or used on the ontological level. Trastour et al. [10, 11] 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 axiomatising implicit knowledge underlying RosettaNet PIP processes.

8. CONCLUSIONS

The scenario discussed in the paper on quoting and purchasing highlights the problems currently observed in RosettaNet collaborations. For example, having suppliers from different countries brings heterogeneities as the partners are likely to use different currencies, different measurement units or different packaging units. Benefits of resolving heterogeneities for the buyer result from decreased costs of purchasing as the best value deals can be selected based on best quotes. The sellers benefit from being able to easier

integrate to the buyer without having to make potentially costly changes to their current integration interfaces.

Our semantic B2B gateway allows a buyer to tackle such heterogeneities in RosettaNet interactions. The solution relies upon a formalised RosettaNet ontology including axiomatised knowledge and rules to resolve data heterogeneities. We showed how to capture definitional facts such as the relation between pounds and kilograms by defining functions in the ontology relating units of measurement. These relations are not specified by RosettaNet.

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. The solutions provided in this paper have potential use in significant portion of the 190 RosettaNet PIP messages.

Acknowledgments

This material is based upon works supported by the Science Foundation Ireland under Grant No. SFI/02/CE1/I131 and No. SFI/04/BR/CS0694. 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.

9. REFERENCES

[1] N. Anicic, N. Ivezic, and A. Jones. An Architecture for Semantic Enterprise Application Integration Standards. In *Interoperability of Enterprise Software and Applications*, pp. 25–34. Springer, 2006.

[2] J. de Bruijn, H. Lausen, A. Polleres, and D. Fensel. The web service modeling language: An overview. In *Proc. of the European Semantic Web Conference*. 2006.

[3] S. Damodaran. B2B integration over the Internet with XML: RosettaNet successes and challenges. In *Proc. of the 13th Int. World Wide Web Conference – Alternate track papers & posters*, pp. 188–195. ACM Press, 2004.

[4] D. Foxvog and C. Bussler. Ontologizing EDI: First Steps and Initial Experience. In *Proc. of the Int. Workshop on Data Engineering Issues in E-Commerce and Services*, pp. 49–58. 2005.

[5] A. Haller, E. Cimpian, A. Mocan, E. Oren, and C. Bussler. WSMX – A Semantic Service-Oriented Architecture. In *Proc. of the 3rd Int. Conf. on Web Services*, pp. 321 – 328. IEEE Computer Society, 2005.

[6] P. Kotinurmi, T. Vitvar, A. Haller, R. Boran, and A. Richardson. Semantic web services enabled B2B integration. In *Proc. of the Int. Workshop on Data Engineering Issues in E-Commerce and Services*. 2006.

[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 *Proc. of the 4th Int. Semantic Web Conference*. 2005.

[8] D. Roman and J. Scicluna. Ontology-based choreography of WSMO services. WSMO final draft v0.3, DERI, 2006. http://www.wsmo.org/TR/d14/v0.3/.

[9] D. Roman, *et al.* Web Service Modeling Ontology. *Applied Ontologies*, 1(1):77 – 106, 2005.

[10] 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.

[11] D. Trastour, C. Preist, and D. Coleman. Using Semantic Web Technology to Enhance Current Business-to-Business Integration Approaches. In *Proc. of the Int. Enterprise Distributed Object Computing Conference*, pp. 222–231. 2003.

[12] M. Zaremba, M. Moran, and T. Haselwanter. WSMX architecture. D13.4 wsmx final draft v0.2, DERI, 2005. http://www.wsmo.org/TR/d13/d13.4/v0.2/.