Towards a complete SCM Ontology - The Case of ontologising RosettaNet

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

Jedrzej Gontarczyk
DERI
National University of Ireland, Galway
jedrzej.gontarczyk@deri.org

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

ABSTRACT

This paper presents a methodology to derive a Supply Chain Management Ontology based on the RosettaNet specification framework. A prototype to mechanically derive a core ontology spanning all new Partner Interface Processes in the RosettaNet framework is developed and its algorithms to reconcile the ontology structure and to generate a proper subsumption hierarchy are presented. We further present how we designed and referenced outer layer ontologies we analysed to be required to resolve the remaining disparities in the core ontology. The resulting ontology framework enables to more easily deal with different message structures in dynamic Business-to-Business collaborations and thus ensures a better interoperability for the partners involved.

Categories and Subject Descriptors

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

Keywords

Supply Chain Management, RosettaNet, ontology engineering, Semantic Web Services

1. INTRODUCTION

Information and communication technologies are increasingly important in the daily operations of organisations. In the current networked business environment, most information systems need to interoperate with other internal and external information systems. Standards, such as RosettaNet[1], UBL [3] or ebXML [16], facilitate Business-to-Business (B2B) integration. These standards support electronic commerce over existing Internet standards and lead to cost and extensibility benefits.

RosettaNet is an industry-driven e-business process standard that defines common inter-company public processes and their associated business documents. The most important component in RosettaNet are Partner Interface Processes (PIPs), containing a specification document, a set of document schemas and message guidelines to help to interpret these schemas. The document schemas of earlier PIPs have been expressed using Document Type Definitions (DTD), leading to expressivity and coherency issues [8, 7].

In this paper we focus on the 50 PIPs which are available as XML schemas (XSD) in August 2007. Although a majority of PIPs used in production systems are still specified using DTDs, all PIPs published after 2004 use XSDs. With the introduction of XSD schemas the RosettaNet framework exhibits a coherent modular design, instead of the monolithic model caused by the use of DTDs. In the latter, the technical dictionaries and general purpose specifications such as metrics, dates, freight codes etc. have been kept internal to each individual PIP. This led to inconsistencies in the standard and caused the adoption to be difficult to maintain if two partners used different versions of the same PIP.

In XSD-based PIPs types are coherently used throughout the RosettaNet standard where each PIP references the same schematic representation for similar information and extends the schema when used in a different context. In its current version RosettaNet recommends in most cases the use of external standards, such as ISO coding standards for countries and currencies. Such information was previously specified in the monolithic model. However, these elements are usually defined as tokenised strings with the values either unspecified or merely syntactically restricted. The interpretation of its usage is left up to the interpretation of the individual implementing company. These issues are a major source of incompatibility in the usage of a PIP, since the relation between types (e.g. tokens) and their semantics is not made explicit. We propose to ontologise the complete RosettaNet specification to form a foundational model for a general purpose Supply Chain Management (SCM) ontology. We present a methodology how to mechanically derive this ontology from the XSD-based PIPs and how semantic relations and a proper ontology structure can be automatically generated. We analyse the usage of different elements throughout the RosettaNet specification and subsequently through our generated core ontology and identify the most likely remaining sources of heterogeneities. We propose a set of outer layer ontologies to resolve these heterogeneities in a dynamic SCM setting where partners can not rely on contractual agreements to restrict the usage of token elements (e.g. an agreement to only use metric types or USD in the quoting).

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

SAC'08 March 16-20, 2007, Fortaleza, Ceará, Brazil

The paper is structured as follows: first we give a brief introduction to the Semantic Web technology stack required to ontologise the RosettaNet specification and position our work to related literature in section 2. We then motivate our solution using an example RosettaNet Quote-to-Cash process in section 3. Section 4 presents the ontology engineering methodology, including the mechanical extraction of a core RosettaNet ontology in section 4.1 and its algorithms to reconcile the ontology structure and to generate a proper subsumption hierarchy. We further present the outer layer ontologies in section 4.2 we analysed to be required to resolve remaining heterogeneities in the core ontology. Section 5 discusses the generalisability of our proposed solution and the issues companies face when applying such a solution. We conclude in section 6.

2. BACKGROUND

In this section we give a short overview of the recent Semantic Web initiatives, standards and languages. We further present how these technologies have already been applied in other research efforts to more expressively define business documents and standards.

2.1 Semantic Web technologies

The Semantic Web idea [2] and its accompanying technologies were introduced to express information not only in natural language or in predominantly syntactical formats such as HTML or XML, but in a way that it can be read and used by software agents, to infer, find and integrate information more easily. Ontologies [10] represent a core pillar of the Semantic Web idea, as they define a set of concepts within a domain and the relationships between those concepts. Although the current RosettaNet specification can be thought of as a lightweight ontology, it is not represented in a language which supports reasoning about the objects within that domain.

Multiple standardisation efforts aim to define a framework and parts of a so called Semantic Web layer cake, including amongst others RDF(s) [13, 4] and OWL [15]. We have chosen the Meta Model offered by the Web Service Modeling Ontology (WSMO) [6] and its accompanying ontology language (WSML) [5] to model and implement the ontology described in this paper. The choice has been made on the fact that we require a more expressive language than RDF(s), but also a framework treating services as first-class citizen in its meta model. However, although we have opted to use WSML, the ontology could theoretically also be defined in OWL.

The Web Service Modeling Language (WSML) [5] is an ontology language offering a human readable syntax, as well as XML and RDF syntaxes for exchanging data between services. WSML clearly separates between conceptual and logical expression syntaxes. The conceptual syntax is used to distinctly model different conceptual aspects of WSMO such as Web services, Ontologies, Goal and Mediators, whereas the logical expression syntax is used for describing additional constraints and axioms. WSML consists of a number of variants based on different logical formalisms. The different variants of the WSML correspond to different levels of logical expressiveness and are both syntactically and semantically layered.

2.2 Related Work

Anicic et al. [1] present how two XML Schema-based automotive standards, AIAG and STAR, are translated from XML to an OWL-based ontology. Their methodology is similar to ours although it uses a different base schema and is formalised in OWL [15]. Our approach further identifies and defines outer layer ontologies, but in overall the efforts are complementary. Foxvog and Bussler [9] describe how EDI X12 can be presented in the WSML, OWL and CycL ontology languages. The work focuses on issues encountered when building a general-purpose B2B ontology. This work provides a basis for mapping EDI concepts to our SCM concepts and could serve as a complementary effort. Trastour et al. [18, 19] augment RosettaNet PIPs with partner-specific DAML+OIL constraints and use agent technologies to automatically propose modifications if the partners use messages differently. Their approach is related to our XSD to WSML transformation methodology, but does not include a rich ontologisation as proposed in our outer layer ontologies. Hepp [12] presents a methodology how to derive an ontology out of an existing product classification standard. Their work is complementary to our efforts and is indeed used in our classification ontology.

3. MOTIVATING EXAMPLE

To show a typical supply chain process, we consider a quote-to-cash process involving a buyer and multiple sellers utilising B2B standards in the communication. Despite using standards, there is often a lot of heterogeneity as the standards can be used differently.

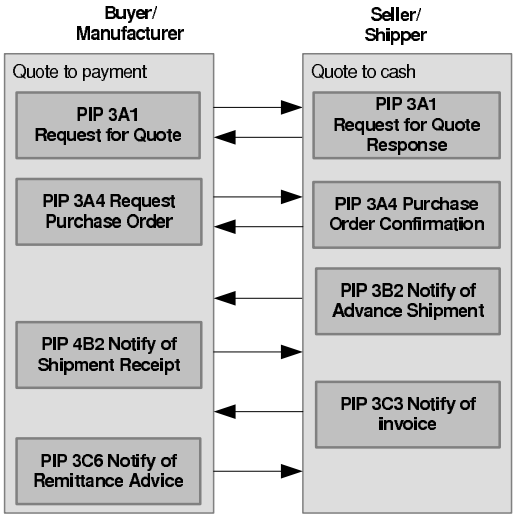

Figure 1: Quote-to-Cash process

Figure 1 shows the complete quote-to-cash process with one partner using RosettaNet PIPs. However, in the quoting phase, there are typically many competing offers. The products offered by different providers can also be substitutable products that are replaceable to the initially quoted products. The quotes and orders can arrive using different measurement units and currencies. The best quote is selected and actual orders are made. The shipment of the products can occur in different lot sizes and the transportation method and equipment used constrain how the products

are shipped. After receiving the products the payment processes are initiated. This typical quote-to-cash process covers many RosettaNet PIPs. These include quoting for price and delivery details (3A1), making the actual order (3A4), shipping the goods (3B2, 4B2) and handling payment (3C3, 3C6).

There are many B2B standards for SCM that specify the needed business messages related to these processes. For instance, RosettaNet PIP 3B2 *Notify of Advance Shipment* carries logically similar information to UBL 2.0 *Despatch Advice*. The information in such messages contains detailed product level contents of the shipment, the delivery time and other shipment information including product identification, shipment quantities and measurements. Listing 1 provides extracts of RosettaNet documents that still use different types in the PIP message for measurement, identification and quoting information. Further, if companies use different standards such as UBL and ebXML, they support yet other ways how to represent logically similar information. This introduces further challenges for the manufacturing company that needs to be able to interpret the multiple representations.

```
<!--RosettaNet example 1-->
<updi:ProductIdentification>
  <udt:GTIN>55566677788899</udt:GTIN>
</updi:ProductIdentification>
<uuom:UnitOfMeasure>KIG</uuom:UnitOfMeasure>
%KIG short for Kilogram
<ume:FinancialAmount>
   <ume:Amount>100</ume:Amount>
   <ucr:Currency>EUR</ucr:Currency>
</ume:FinancialAmount>

<!--RosettaNet example 2-->
<updi:ProductIdentification>
   <updi:ProductName>hammer</updi:ProductName>
      <ulc:AlternativeIdentifier>
         <ulc:Authority>Buyer</ulc:Authority>
         <ulc:Identifier>6578489</ulc:Identifier>
      </ulc:AlternativeIdentifier>
   </updi:ProductIdentification>
<uuom:UnitOfMeasure>USP</uuom:UnitOfMeasure>
%USP short for US pound
<ume:FinancialAmount>
   <ume:Amount>136</ume:Amount>
   <ucr:Currency>USD</ucr:Currency>
</ume:FinancialAmount>
```

Listing 1: Example RosettaNet message extracts

4. A COMPLETE SCM ONTOLOGY

In the following section we discuss the ontology engineering methodology we followed to create the RosettaNet core ontology. We describe its main concepts and the key modelling decisions taken and show how we approached the modelling of the outer ontology layers, resolving the remaining source of heterogeneities in RosettaNet as discussed in the motivating example.

The process of developing a complete Supply Chain ontology from RosettaNet schemas is carried out in two steps. (1) We mechanically derive a core ontology, being a direct translation from XSD to WSML including a reconciliation phase to hierarchically structure the ontology and to add a proper subsumption hierarchy; (2) we analyse the RosettaNet specification and identify remaining sources of heterogeneity in order to model and reference richly axiomatised ontologies, forming the outer layer in our ontological framework.

4.1 Core layer

The consequent structure of the RosettaNet specification allowed us to develop a methodology and tool for mechanically deriving a consistent, lightweight core RosettaNet ontology. The prototype application allows us to update the base ontology by applying changes to the application code rather than to every single ontology fragment whenever a correction is made to the RosettaNet specification (XSD files) or a new feature is added.

4.1.1 Extract the Core ontology

The process of creating the base ontology is as follows:

1. The document header of the XSD file is parsed to convert the list of referred namespaces directly to namespaces in WSML.

2. The ontology itself is created and identified with the the name of the originating XSD file. The XSD version numbers become part of the target namespace definition in order to add semantic relations between different ontology versions at a later stage.

3. The schema annotation nodes are transformed into non-functional properties in the ontology.

4. Next, all stand-alone types (elements with no child nodes referencing another type) are translated to subconcepts of their referenced types. All stand-alone types are of type <`xs:element`>, whereas all top-level <`xs:element`> tags represent stand-alone types. All nodes referencing the namespace prefix *tns* belong to the local namespace and can be omitted in WSML.

5. In a next step, all the remaining <`xs:simpleType`> and <`xs:complexType`> are handled.

   For all <`xs:simpleType`> definitions used to create a new type based on restricting one of the XSD built-in types, the mapping results in the creation of a concept in the ontology with a type of the built-in type in WSML. An axiom is added for possible value restrictions. However, WSML does not support pattern constraints, thus we omit them in the translation. If the <`xs:simpleType`> definition is based on a list of other simpleTypes, the mapping results in the creation of a concept with an attribute resulting from the transformation of the list.

   Complex types always map to a concept in WSML. Sub elements with a simple built-in type are mapped to attributes with the same built-in type. Sub elements with simple types that are not WSML built-ins are mapped to attributes with the type of the mapped simple type definition. A sub element that itself is a complex type leads first to the creation of a corresponding concept as shown in listing 2. The XSD sequence element, specifying that the child elements must appear in a sequence, is ignored in the translation to its ontological schema, since structural restrictions are not the scope of an ontology. XSD choice elements on the other hand are handled, since it restricts instances to be of a particular type of the list of types. We add a supporting axiom to the ontology.

```
<xs:complexContent>
 <xs:extension base="uat:IdentifierType">
  <xs:sequence>
    <xs:element name="ProductName" type="xs:string"
        minOccurs="0">
    <xs:element name="Revision" type="xs:string"
        minOccurs="0">
  </xs:sequence>
 </xs:extension>
</xs:complexContent>

hasIdentifierType ofType extIdentifierType

concept extIdentifierType subConceptOf uat#IdentifierType
    ProductName ofType (0 1) _string
    Revision ofType (0 1) _string
```

Listing 2: Complex extension type and its WSML representation.

6. The final step concerns the reconciliation of the type hierarchy. Although the type hierarchy in the current RosettaNet specification is mostly coherent and shares similar types throughout different PIPs, in several cases some types not only share the name (in a different namespace), but also shares some elements. In such cases we merged the types, determining the common elements and creating a parent concept, placing it in the *Universal* namespace. With this concept, the overlapping type definitions in the originating types are removed and added as a sub-concept of the newly created concept in the *Universal* namespace. This operation greatly reduced the size of the specification overall, while making it cleaner and more expressive (subsumption relations) in its type hierarchy.

Following this simple transformation methodology we can derive an extensive ontological framework. However, attributed to the modular structure of the schema, the advantages of a mechanically derived ontology without adding semantic relations is limited. Therefore our transformation methodology includes the following steps to structure the ontology documents and to automatically add a proper subsumption hierarchy.

4.1.2 Reconciliation of the core ontology structure

RosettaNet's original structure is based on namespaces. Every data type is accessible through the namespace of the schema document it belongs to. Moreover, schemas differ by level of their accessibility.

- Schemas with general accessibility throughout all PIPs belong to a class *Universal*.
- PIPs are organised into eight clusters, denoted by numbers (e.g. PIP**3**xx, PIP**4**xx), which are further broken down into segments, denoted by letters (e.g. PIP3**A**x, PIP4**B**x). For example, cluster 3 deals with *Order Management*; it is divided into four segments including for example PIP3A *Quote and Order Entry*. Accordingly, schemas only shared between the members of a cluster belong to groups called *procurement*, *salesreporting* etc. and *shared* classes, being shared by PIPs in the whole segment.
- Schemas which can be accessed only by one PIP belong to the *Interchange* class (schemas of this class are always used to define package-specific data types).

Although this grouping ensures a modular structure in RosettaNet, it is not properly designed. First, the sharing of schemas is achieved by including copies of every file among the packages which as pointed out earlier can lead to inconsistencies in the specification. Secondly, schemas are a top-level structure and not grouped in any logical way (other than in filesystem directories), making it impossible to query for schemas belonging to a similar cluster, segment or package.

In the ontologisation process we addressed these problems so that one can query the ontology for cluster, segment, package and PIP memberships. The process we followed is as follows:

- First we regrouped the directory structure so that no file exists in more than one copy.
- All ontological schemas of the same level of accessibility are assigned to three top-level groups (universal, domain, interchange). Those groups further separate schemas by their cluster/segment/package membership.
- Next, a domain group containing a procurement and shared ontology with schemas/ontologies shared throughout all PIPs and a cluster ontology including groups of schemas used within different RosettaNet clusters, such as *Manufacturing* and *Logistics* is added.
- Finally, a *Universal* ontology referencing all schemas belonging to the universal namespace, and an *Interchange* ontology containing the interchange document ontologies from the different PIPs are created.

4.1.3 Deriving a subsumption hierarchy

While parsing the Document Object Model tree we apply a simple language processing technique to identify compounded words. RosettaNet uses compounded nouns throughout the specification and many of them share similar types. Thus, we search for the occurrence of its lexemes (the start of a new lexeme in the specification can be easily identified by its capitalisation) as part of a compounded noun in the type name. If we find multiple types, sharing a lexeme in its compounded names, we check their referenced types. If the element references a similar type, a top level concept with the respective part of the compound noun is created and subclass relations to all occurrences of the compounded word are added. Listing 3 shows an example of such a generated concept hierarchy. RosettaNet includes for example multiple quantity types, such as *OrderQuantity*, *ProductQuantity*, *ForecastQuantity*, *LeadTimeQuantity* etc. By applying this simple algorithm, only taking compounds into account where *Quantity* is occurring at the right-hand we derive ten elements with the same type and create the concepts as shown in listing 3.

```
concept Quantity
    hasNumericalValue ofType _float

concept OrderQuantity subConceptOf Quantity
concept ProductQuantity subConceptOf Quantity
concept ForecastQuantity subConceptOf Quantity
...
```

Listing 3: Concept hierarchy derived by lexical analysis

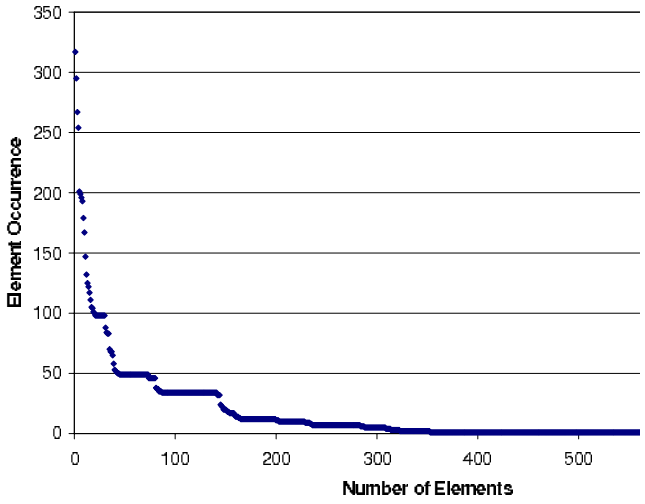

Figure 2: Distribution of element occurrence

4.2 Outer layers

RosettaNet is a Supply Chain Management standard defining the message exchange patterns and the document content for Order Management, Quoting, Financial Review, Inventory Management etc. in the information technology, electronic components, semiconductor manufacturing, telecommunications, and logistics industries. However, information exchanged in PIP documents spans multiple knowledge domains. Since the documents are interpreted by different parties in the collaboration, all information has to be uniformly understood. Since RosettaNet does not and can not model all domain knowledge, such as the actual product and partner identification, measurement types or currency types, other specifications (ontologies) have to ensure the uniform understanding of the elements within a message.

RosettaNet publishes only guidelines which standards to use, but does not reference or integrate ontologies to homogenise their usage. Table 4.2 analyses the element occurrence in all of the fifty RosettaNet PIPs currently available as XSD schemas. The table distinguishes between mandatory (left) and optional element occurrences (right) in the schema.

ProductIdentification	317
UnitOfMeasure	295
FinancialAmount	267
ProductQuantity	254
MonetaryAmount	201
PartnerIdentification	199
ProcessRoleIdentifier	196
AlternativeIdentifier	193
Identifier	179
DatePeriod	167

Table 1: Occurrence of elements considering cardinalities

ProductQuantity	490
DatePeriod	469
ProductIdentification	444
FinancialAmount	381
UnitOfMeasure	379
ContactInformation	353
BusinessDoc.Ref.	349
MonetaryAmount	333
PercentAmount	288
Measure	252

Table 2: Occurrence of elements omitting cardinalities

The analysis shows power law distribution characteristics (cf. figure 2) with few elements used often and many elements used only few times.

The rank of the individual element in the two datasets is quite similar. The coefficient of variation (c_v) of the ranks of the same elements in the dataset considering cardinalities is $c_v = 0.2588$ compared to $c_v = 0.2459$ in the dataset omitting the element cardinalities. Although the ranking result are almost similar, it is safe to assume that the element occurrence taking cardinalities under consideration is closer to the actual usage of the elements in individual XML instances. In fact a positive cardinality still does not guarantee the actual usage of an element, because its usage context might be of zero cardinality. However, the schema occurrences of mandatory elements is a reliable indicator of the actual usage and thus we will analyse the top ranked elements taking their cardinality into account.

Based on this metric we start to design the outer layers of a SCM ontology. The element with the highest number of occurrences, *ProductIdentification*, similarly to the sixth most used element, the *PartnerIdentification*, references one of the following identifier types: `uat:identifiertype`, `ulc:AlternativeIdentifier`, `udt:GTIN`, `udt:DUNS`, `udt:DUNSPlus4` and `udt:GLN`. The role of these identifiers is to describe products (GTIN), companies (DUNS) and locations (GLN or DUNSPlus4 for company location) uniquely. When, for example, ordering a hammer, it is easier to refer to an identifier, such as the 14-digit GTIN "55566677788899", than specifying "Tool, hammer, handtool, Fiskars, ..." in every business message. Alternative identifiers can also be used, typically for buyer identification. Using differing identification schemas creates a mapping challenge where ontologies can help to state similarity between different identifiers for a product. Since an n-to-n mapping between identifiers is unfeasible, we propose to map to an existing product classification such as the eCl@ss classification (code "AAA374002" for hammer). By specifying this semantic detail and referring to the existing eClassOwl-ontology[2] [12], it is possible to provide information on similar products, such as hammers produced by other manufacturers. Further benefits materialise when respective UN/SPSC classifications are mapped to eCl@ss categories. This enables companies that map their identifier codes to a category to provide substitutable products and increase their chance of realising sales. Similarly, the current location identifiers can be mapped to location ontologies and thus provide routing services for the shipment companies. Currently used dummy identifiers do not help to make any intelligent decisions.

The second most references element, *UnitOfMeasureType*, extends the *UnitOfMeasureContentType* with a list of 367 measurement types. They are directly modelled and constrained in RosettaNet as tokenised strings. All the inherent relations between the individual tokens are left unspecified. First, we identified for each tokenised string in the XSD Schema its unit type class membership in the Suggested Upper Merged Ontology (SUMO) [14]. SUMO is a richly axiomatised formal ontology created by the merger of multiple existing upper-level ontologies. SUMO is divided into eleven sections whose interdependencies are carefully documented. We are mostly interested in classes from the base ontology, numeric and measurement layer. Other parts of the ontology include among others, temporal, process and class theories. All unit types in the 367 token values can be related to the *PhysicalQuantity* class in SUMO, which

[2]See http://ontologies.deri.org/eclass/5.1/#C_AAA374002

itself subclasses *ConstantQuantity*, *FunctionQuantity* and *UnitOfMeasure*. By using SUMO we derive foundational relations, such as the equivalence of 1 litre to 1 cubic decimetre and the relation that 4.54609 litres are equal to 1 *UnitedKingdomGallon*, all facts defined in the axiom schemata of SUMO. Listing 4 shows the concept definitions in the ontology and its related classes in SUMO. *PoundMass* and *Kilogram* in SUMO are second level subclasses of *ProductQuantity*.

```
concept ProductQuantity subConceptOf sumo#ProductQuantity
concept PoundMass subConceptOf sumo#PoundMass
concept Kilogram subConceptOf sumo#Kilogram
```

Listing 4: *UnitOfMeasureTypes* in the RosettaNet ontology

Next, we identified common types in the tokens and modelled them as concepts in the ontology. Examples of similar tokens are a *10 Kilogram Drum*, a *100 Pound Drum* and a *15 Kilogram Drum*. Listing 5 shows the first two tokens in its XSD Schema definition and listing 6 shows its representation in our ontology. We identified the concept *Drum* as being member of a *FluidContainer* in SUMO and it inherits similarly to all other converted unit types the *hasTokenValue*, *hasUnitSize* and *hasUnitType* attributes from its parent concept (*Quantity*).

```
<xs:simpleType name=
  "UnitOfMeasureContentType">
 <xs:restriction base="xs:token">
  <xs:enumeration value="1KD">
   <xs:annotation>
    <xs:appinfo>
     <urss:Definition>
      10 Kilogram Drum.
     </urss:Definition>
    </xs:appinfo>
   </xs:annotation>
  </xs:enumeration>
  <xs:enumeration value="1PD">
   <xs:annotation>
    <xs:appinfo>
     <urss:Definition>
      100 Pound Drum.
     </urss:Definition>
    </xs:appinfo>
   </xs:annotation>
  </xs:enumeration>
 </xs:restriction>
</xs:simpleType>
```

Listing 5: UnitOfMeasureTypes tokens in XSD Schema

```
concept ProductQuantity
        subConceptOf sumo#
        ProductQuantity
    hasTokenValue ofType _string
    hasUnitQuota ofType _float
    hasUnitType ofType
            ProductQuantity

concept Drum subConceptOf {
        sumo#FluidContainer,
        ProductQuantity}

instance _10KilogramDrum
        memberOf Drum
    hasTokenValue hasValue "1KD"
    hasUnitQuota hasValue 10
    hasUnitType hasValue Kilogram

instance _100PoundDrum
        memberOf Drum
    hasTokenValue hasValue "1PD"
    hasUnitQuota hasValue 100
    hasUnitType hasValue
            PoundMass
```

Listing 6: ... and in the RosettaNet ontology

This style of modelling allows us to further include semantic relations between instances of the same unit type concept. To define the numerical dependencies between different *UnitOfMeasureContentType* we add equivalence relations similar to the one shown in listing 7. It states that a *Quantity* instance with a certain amount $?z$ of "100 Pound Drum" unit types equals 4.5359237 times 10 Kilogram Drums. Since we have derived a subsumption hierarchy in the transformation step described in section 4.1.3, this axiom applies to all sub-classes of *Quantities*, such as *ProductQuantity*, the fourth most used type in the RosettaNet schema.

The third most used element *FinancialAmount* references a float type for the currency amount and the *ucr:Currency* type, which references the ISO 4217 standard on currencies[3]. The currency identifier is supposed to be entered in the tokenised string. Similarly to the *MeasurementUnitType* ontology we identify the class membership of each currency in SUMO, and model all currencies as concepts. However, we can not directly include the semantic relation between the currencies in the ontology, because naturally exchange rates are constantly changing. However, we can relate every currency to two base currencies, the Euro and Dollar and define a service invocation to retrieve the actual exchange rate. We have presented a solution to that problem in an earlier work [11].

```
axiom _1PD1KDDependency
    definedBy
        ?x memberOf Quantity and ?x[hasUnitType hasValue
                _100PoundDrum] and ?y[hasNumericalValue hasValue ?z]
        equivalent
        ?x memberOf Quantity and ?x[hasUnitType hasValue
                _10KilogramDrum] and ?y[hasNumericalValue hasValue
                wsml#numericMultiply(?z1,?z,4.5359237)].
```

Listing 7: Equivalence relation between 100 Pound Drum and 10 Kilogram Drum

The *MonetaryAmount* is related to the *FinancialAmount* type and only specifies whether the amount is a debit or credit. Both types do not cause any heterogeneities and are omitted in our ontologisation.

The *ProcessRoleIdentifier* references 56 tokens which describe the role a partner plays in the collaboration. For every PIP there are a few possible roles specified. However, such information is of non-functional nature and typically straightforwardly interpreted with all partners.

RosettaNet supports different ways to specify a certain *datePeriod*. One can either use starting and ending times or the start time and duration. With ontologies supporting both is straightforward as the logical connections can be easily represented. We reuse a Date and Time ontology defined in [17] to explicitly relate different ways of modelling time.

5. DISCUSSION

We defined outer layer ontologies for the top most occurrences of elements in the RosettaNet schema. Based on our occurrence analysis, the sum of the occurrences of the top ten most referenced elements including their leaf-nodes referenced (in case they are not leaf-nodes themselves) in all fifty XSD-based PIPs, amounts for 23% of the total number of types. Thus, the semantic relations defined in the outer layer ontologies presented above, can solve a significant number of heterogeneities between partners in a dynamic B2B setting.

The RosettaNet specification includes currently 933 different types. To eliminate all possible sources of heterogeneity one would have to analyse all types and define semantic relations between all ill-defined types. The effort of creating such a complete SCM ontology is out of scope of a research effort. However, it is also not realistic to be undertaken by

[3]See http://www.bsi-global.com/en/Standards-and-Publications/Industry-Sectors/Services/BSI-Currency-Code-Service/

any business partner on its own. Every organisation participating in a RosettaNet collaboration will have to identify their most likely sources of heterogeneity and apply an ontologisation effort similarly to the one we have described in section 4.2. The effort of explicitly defining mappings is cost-effective only to organisations that need to collaborate with multiple partners. Organisations using only a small part of existing B2B standards and implement it only with a limited number of partners can more efficiently encode mappings in the B2B adapters directly. However, in particular companies with multiple partner collaborations who are not able to dictate their standards to partners along the Supply Chain can benefit by utilising the outer layer ontologies as described in section 4.2. Such ontologies only need to be defined once (preferably by the business community), but can be reused in many business interactions as opposed to hard-coded rules implemented in every bi-directional collaboration.

6. CONCLUSIONS

In this paper we presented a methodology to derive a Supply Chain Ontology based on the RosettaNet specification. We extend the ontology with general purpose ontologies in an attempt to resolve heterogeneities, not structurally and semantically covered by the RosettaNet specification. We developed a prototype to mechanically derive a core ontology spanning all the XSD schema PIPs, implementing algorithms to reconcile the ontology structure and to generate a proper subsumption hierarchy. We analysed the occurrence of all 933 types used throughout the 50 XSD-based RosettaNet PIPs and presented outer layer ontologies to account for the ten most referenced elements. We explicitly defined the inherent relations between the individual objects of these types. The solution presented for the outer layer ontologies amounts for 23% of the total number of types. Thus, the semantic relations defined in our SCM ontology can already solve a significant portion of heterogeneities in messages exchanged between partners in a dynamic B2B setting.

We envisage further evaluation of our ontology by collecting actual RosettaNet instances and translating them with our prototype application. This will allow us to compare, if the amount of heterogeneities as we analysed based on the element occurrence frequency at schema level accounts for a similar amount of heterogeneities on its instance level.

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).

7. REFERENCES

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

[2] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, 284(5):34–43, 2001.

[3] J. Bosak, T. McGrath, and G. K. Holman. Universal business language v2.0, Dec. 2006.

[4] D. Brickley and R. V. Guha. RDF Vocabulary Description Language 1.0: RDF Schema. W3C Recommendation, 10 February 2004. Available from `http://www.w3.org/TR/rdf-schema/`.

[5] J. de Bruijn, *et al.* The Web Service Modeling Language WSML. WSML Final Draft D16.1v0.2, DERI, Oct. 2005. From `http://www.wsmo.org/TR/d16/d16.1/v0.21/`.

[6] J. de Bruijn, *et al.* Web Service Modeling Ontology (WSMO). Member submission, W3C, 2005. Available from: `http://www.w3.org/Submission/WSMO/`.

[7] S. Damodaran. B2B integration over the Internet with XML: RosettaNet successes and challenges. In *Proceedings of the 13th International World Wide Web Conference on Alternate track papers & posters*, pp. 188–195. ACM Press, New York, NY, USA, 2004.

[8] S. Damodaran. RosettaNet: Adoption Brings New Problems, New Solutions. In *Proceedings of the XML 2005 Conference*, pp. 1–14. Atlanta, USA, 2005.

[9] D. Foxvog and C. Bussler. Ontologizing EDI Semantics. In *Proceedings of the Workshop on Ontologising Industrial Standards*, pp. 301–311. Springer, Tucson, AZ, USA, 2006.

[10] T. R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199–229, 1993.

[11] A. Haller, P. Kotinurmi, T. Vitvar, and E. Oren. Handling heterogeneity in RosettaNet messages. In *Proceedings of the 22nd Annual ACM Symposium on Applied Computing (SAC)*, pp. 1368 – 1374. ACM, Seoul, Korea, Mar. 2007.

[12] M. Hepp. A methodology for deriving OWL ontologies from products and services categorization standards. In *Proceedings of the 13th European Conference on Information Systems (ECIS2005)*, pp. 1–12. 2005b.

[13] G. Klyne and J. J. Carroll. Resource description framework (RDF): Concepts and abstract syntax. W3C Recommendation, 10 February 2004. Available from `http://www.w3.org/TR/rdf-concepts/`.

[14] I. Niles and A. Pease. Towards a standard upper ontology. In *Proceedings of the International Conference on Formal Ontology in Information Systems*, pp. 2–9. New York, NY, USA, 2001.

[15] P. F. Patel-Schneider, P. Hayes, and I. Horrocks. OWL web ontology language semantics and abstract syntax. W3C Recommendation, 10 February 2004. Available from `http://www.w3.org/TR/owl-semantics/`.

[16] 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.

[17] M. Stollberg, *et al.* Wsmo use case "virtual travel agency". WSMO Working Draft D3.3 v0.1, DERI, 2004. `http://www.wsmo.org/2004/d3/d3.3/v0.1/`.

[18] 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.

[19] 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*, pp. 222–231. IEEE Computer Society, 2003.