Mapping between heterogeneous XML and OWL transaction representations in B2B integration

Jorge Cardoso[1] and Christoph Bussler[2]

[1] Corresponding author
CISUC/Departamento de Engenharia Informática, Universidade de Coimbra, Portugal.
E-mail address:jcardoso@dei.uc.pt
Tel: +351 239 790 000; fax: +351 239 701 266.
[2] Saba Software, Inc., USA.
E-mail address:chbussler@aol.com

Abstract. XML-based standards have been widely used to enable and ease Business-to-Business (B2B) integration. Examples of standards include cXML, CIDX and ebXML. While these XML-based standards are syntactic, contemporary organizations have available new means to structure their internal data representations using semantic descriptions, such as RDF(S) and OWL. This scenario poses an interesting challenge: "How to reconcile external XML-based standards and internal OWL-based representations in B2B integration scenarios?" In this paper, we present a conceptual approach, and its implementation, to integrate external syntactic data representations with organizational internal semantic data representations by using the notion of heterogeneous mappings which are established between the two types of representations. The application developed, B2BISS, enables an effective management of mappings. As the number of mappings stored in the repository increases over time, organizations can gradually rely on a semi-automatic to automatic B2B integration.

Keywords: Integration; ontology; OWL; XML; schema

1 Introduction

Global economies are increasingly becoming networked. The notion of value chains [39], value nets [35], b-Webs [45], and value networks [36] as concepts and tools have been used to understand and analyze networked industries. They are useful instruments for portraying the interconnection of operations, players and transactions. Many industries now exhibit strong co-operative behavior with inter-firm relationships having a significant role in strategic performance. The existence of business chains, nets, b-Webs or networks leads invariably to the necessity of developing Business-to-Business (B2B) solutions for integration [32]. The objective of the integration is to ease the management of transaction-based interactions between business players that are part of a network to cut costs, increase revenues, and improve time-to-market [11, 5, 48].

Nowadays, one simple solution that organizations are adopting to reach business network integration is to rely on the use of XML-based domain specific standards to exchange transactions [8]. Examples of well-known standards include cXML (Commerce eXtensible Markup Language), CIDX (Chemical Industry Data Exchange), ebXML (Electronic Business using eXtensible Markup Language), XML/EDIFACT, papiNet, PIDX (Petroleum Industry Data Exchange), and xCBL (XML Common Business Library)[33].

While XML-based standards allow data exchange between networked businesses, they do not guarantee the interoperability of systems. XML only provides syntax to structure the data exchanged in B2B settings. If on one hand external XML-based standards are syntactic [3], with the adoption of semantics to make data explicit, organizations are considering shifting from a syntactic representation level to a semantic one [49, 10]. By using semantic domain models based on ontologies (e.g. RDF(S) or OWL [1]) enterprises acquire several benefits, such as the ability to perform inference on knowledge bases and the capacity to share domain models to easily exchange and integrate information.

The networked economy [23] requires accounting for the nature of alliances, the technical infrastructures of players, and the data exchanged in business networks. Once a network is designed, it is possible to analyze the model to identify gaps. Once gaps are known, implementation plans can be prepared to close them. One important gap to close for achieving a stronger B2B integration is the lack of a common understanding between external standards and internal organizational data representations [30]. Developers are still faced with the problem of understanding the meaning of the information represented in XML-based standards and establish a correspondence or mapping to the internal OWL-based data representation of organizations [34, 48]. The manual software coding of mappings between data models is a time consuming task with high costs.

WSML, WSMO and WSMX [50] use semantically annotated services to promote B2B integration by using data and process integration. Transactions can be semantically enriched using the WSML ontology language. The infrastructure enables domain experts to create mappings between XML-based standards and WSMO ontologies. The mappings are represented in an abstract ontology mapping language. While this research provides an important theoretical contribution, it reveals a few limitations when it needs to be transposed to real-world industries settings. On the one hand, the use of non-standard languages to establish mappings may be considered a penalty for not allowing its sharing across an industry. The use of more well-established languages and tools (such as OWL and XSLT languages and parsers) enables an easier adoption by organizations. On the other hand, the created mappings cannot be reused. This means that for each standard, an its variations, used to represent transactions, a new set of mappings needs to be created. A repository of shared mappings that can be reused overtime would increase the adoption of the approach. Finally, a more responsive and customized system that would automatically classify incoming transactions and requested human involvement when new mappings were needed to fully complete a transformation from instances of one model to instance of the other model would be more adequate in real-world settings.

In this paper we describe a conceptual approach that allows organizations to participate in B2B networks using XML-based syntactic standards to support external transactions while structuring their internal data representations semantically using semantic languages such as OWL. In our approach, partners and suppliers can freely exchange syntactic XML-based transactions. Once an organization receives a syntactic external transaction it is allowed to create an heterogeneous mapping between its elements and the concepts of an internal semantic model (i.e., an ontology) that describes the organization's domain. The conceptual approach to manage heterogeneous mappings has been implemented with the B2BISS system. B2BISS tackles the problem of *information integration* since it fosters aspects such as linguistic and semantic differences to be reconciliated among disparate data representations [38, 11]. The system allows organizations to dramatically decrease the time and cost of integration by providing a flexible and easy to use graphical tool. Such a flexible infrastructure is instrumental for a rapid and cost-effective B2B integration [37].

The remainder of the paper is structured as follows. The second section presents a B2B integration scenario that illustrates the challenge of integrating external XML-based standards and internal OWL-based data representations. Section 3 enumerates the challenges, approaches and our solution to the problem of syntactic (external) to semantic (internal) B2B integration. Section 4 introduces the B2BISS system which enables a (semi-)automatic integration and relies on the notion of managing and reusing heterogeneous mappings. Section 5 describes five organizational scenarios involving the use of the B2BISS system. Section 6 presents a use case. It describes how B2BISS can be implemented and used within the SAP NetWeaver Process Integration infrastructure. Section 7 presents the related work in this area. Finally, section 8 closes the manuscript with our conclusions.

2 B2B integration scenario

Organizations implement B2B infrastructures to allow a seamless interaction with several business partners that include suppliers, distributors, exporters, and retailers (see Figure 1). B2B integration "is not just the buying and selling of goods and services, but also servicing customers, collaborating with business partners, and conducting electronic transactions within an organization" [47]. Drivers for B2B network integration include the need for disintermediation, to reduce inventory cycle time, to optimize business processes, and to use various distribution channels. On the other hand, barriers to integration are related to high technology costs, complex technology (OWL, cXML, ebXML, etc), and confidentiality and privacy concerns.

One very important aspect that needs to be recognized is that different organizations use different internal semantics to describe their products, part

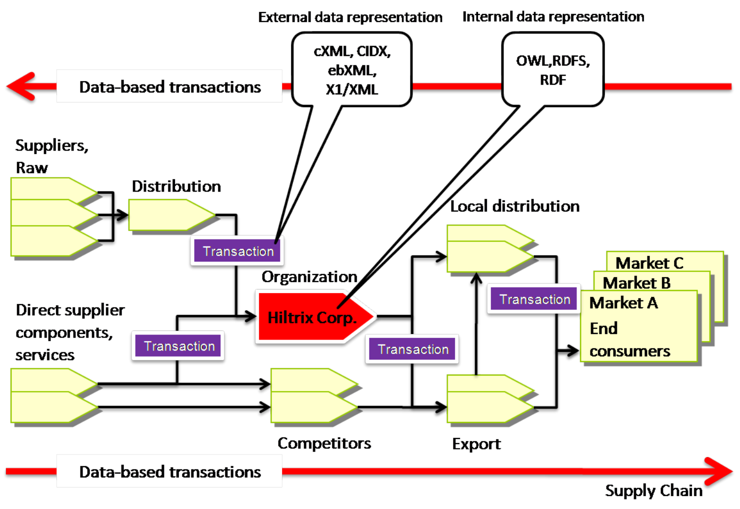


Fig. 1. Transactions in business networks

numbers, invoices, and purchase orders. Therefore, when organizations are collaborating in business networks they have to understand the external data representation of the standards adopted to represent transactions. Then, when an organization receives a transaction, it needs to map its data to its own internal data representation. In Figure 1 it is possible to find two types of data representations: external and internal. The central organization, named Hiltrix Corp., uses ontologies for its data model while it exchanges transactions with suppliers and customers using XML-based standards.

2.1 External data representations

External data representations are used with partners to exchange transactions. In general, every industry develops standards in order for companies to use transactions to communicate with each other. With the advent of the eXtensible Markup Language (XML), it became easier to define and standardize the contents of B2B transactions. XML is a highly flexible, ubiquitous data format, designed for multiple representations.

In progress of time, B2B solutions went through an evolutionary path from monolithic and proprietary standards (e.g. TRADACOMS in UK, ANSI ASC X.12 in U.S.A and UN/EDIFACT in the United Nations) towards flexible and standardized XML-based stacks covering the requirements from different industries [30]. Examples of existing standards to communicate with partners include cXML, CIDX, ebXML, XML/EDIFACT, papiNet, PIDX, and xCBL [8, 6].

XML-based B2B standards generally define a common agreement among business partners on transactions for inter-organization integration. In our approach, we do not make any restriction on the structure of XML transactions exchanged between organizations. As illustrated in Figure 1, organizations can rely on the use of well-known standards such as cXML and CIDX or use proprietary solutions (represented with label X1/XML). For the examples illustrated in this paper, we use the cXML standard since it is the most widely adopted B2B protocol. It is intended to exchange transactions between procurement applications, buyers, suppliers and e-commerce hubs. Nowadays, this protocol is supported by more than fifty international players, including Visa, AMD, Cisco Systems and GM. Some of the procurement systems that implement cXML are the Ariba's eCommerce platform, Peregrine Systems and Oracle Exchange platform. These systems allow users to search and locate buyers and sellers of goods and services.

Listing 3 illustrates a simplified example of an cXML purchase order [13]. The transaction orders 10 units of Poland Spring water at \$1.20 each, 20 units of Scottex tissues at \$1.05, and 30 eight pack (P8[3]) of Minute Maid Frozen juices at \$4.55 each pack. The UNSPSC [22] classification code for Water is 50202301, for Tissues is 12352206 and the code for Frozen juices is 50202303. The total order amount is of \$66.80 and it is to be shipped and billed to Aldo Corp. located in Sunnyvale, California.

2.2 Internal data representations

On the other hand, *internal data representations* are only visible inside organizations. They do not have to be directly interoperable with external data representations (i.e., transactions) as they are used for integrating internal applications such as ERP and CRM systems. Modern approaches involve using semantic representations for organizational data assets to achieve a better description of data. Emerging Semantic Web technologies, such as ontologies, can play an important role in this scenario [12, 18]. Ontologies are a formal and explicit specification of a shared conceptualization which can enable semantic interoperability. Due to the widespread importance of integration for intra- and inter-business networks, the research community has tackled this problem and developed semantic standards such as the Resource Description Framework (RDF) and the Web Ontology Language (OWL) [1]. RDF and OWL standards enable the Web to be a global infrastructure for sharing both documents and data, which makes searching and reusing information easier and more reliable. Figure 2 and Listing 4 illustrate a simplified version of the internal data representation of purchasing orders received by the Hiltrix Corporation described using the ontology language OWL The PurchasingOrder ontology has three classes, Order, Part and Manufacturer. It also has three object properties: hasManufacturer, hasPart and hasItem. The property hasPart is an inverse property of hasManufacturer. Several data properties are present: type, classification, cost, quantity, packaging, and www.

[3] As an example of the elements of the purchase order, the tag UnitOfMeasure of the cXML specification describes how the product is packaged or shipped. It must conform to UN/CEFACT Unit of Measure Common Codes. P8 stands for an eight pack packaging. For a list of UN/CEFACT codes, see www.unetrades.net.

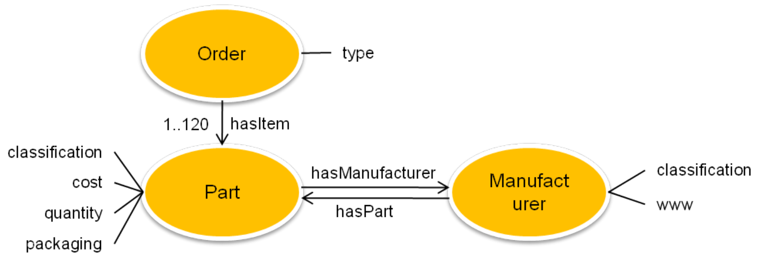


Fig. 2. Simplified illustration of the Hitrix Corp. internal purchasing order ontology

The objective of the work presented in this paper is to propose a solution to integrate the external data representation from Listing 3 with the internal data representation of Listing 4 while having as requirement the minimization of the cost and time needed to carry out their integration. In other words, the goal is to integrate XML and OWL-based data representations.

2.3 XML and OWL data representation differences

The differentiation of external and internal data representation formats results in a set of challenges that need to be addressed in order to enable their integration. Previous research has studied the differences between XML schema and ontology representation languages [27, 21]. The most important differences between the two data representations which are relevant to consider for our work are the following:

– XML provides a rich syntax and structure definitions for data modeling, and it is generally used to define document structures. On the other hand, OWL provides semantics for knowledge domain modeling (e.g. healthcare, bioinformatics, information systems integration).
– The data model of XML describes a node labeled tree, while the data model of OWL is a graph build on RDF triples (relating a subject, a predicate, and an object).
– Since XML describes a labeled tree, the meaning of nested tags is implicit and domain dependent. A nested tag can represent a "part-of" relationship or a "subtype-of" relationship. In OWL, these types of relationships are made explicit with the notion of "class", "subclass" and "property".
– In XML, two distinct nodes with the same name can co-exist at different levels in the labeled tree. This is not allowed in OWL. Every resource needs a unique identifier. Therefore, instances of classes need to obtain an automatically generated identifier.
– XML allows the definition of sequences to describe the order between elements. OWL does not support or impose any order on OWL properties.
– XML only supports inheritance using type derivation by extension or restriction. On the other hand, OWL supports multiple-inheritance. OWL also supports inheritance on properties.

– OWL provides simple logics on relationships for reasoning such as transitivity, disjunction, conjunction and symmetry.

In the view of the fact that our work aims at mapping external XML transactions into an existing internal OWL-based data representation, the aspects which are more important to consider are the ones specific to XML. Each XML element needs to be mapped to a suitable OWL concept.

Since XML does not allow capturing logical constructs such as conjunction, disjunction, negation, existential and universal quantifiers, mappings of this type cannot be theorized from XML to OWL. In XML, semantics are most often "hidden" and not explicitly described in data representations, residing in developers' minds or domain standard specifications. Nonetheless, while they are not used in XML, those logical operators are used by organizations (such as Hiltrix Corp. from Figure 1) which use ontologies to interrelate concepts and describe domain knowledge. To create mappings, users need to inspect "hidden" or implicit semantics of an XML external data representation and establish a mapping to an internal ontological concept. As such, "hidden" semantics and relationships in XML are made visible in OWL through the use of mappings. While the solution presented is simple, the management of heterogeneous mappings from an XML data representation to an ontological data representation enables organizations to extend the expressiveness of XML entities by associating them with the elements of a conceptualized domain.

3 Challenges, approaches and solution

The base problems inherent to the development of solutions for B2B integration that might arise due to the heterogeneity of external and internal data representations are already known within the distributed database community [26]. Nonetheless, as organizations use new representation languages, solutions need to consider the new particularities introduced.

3.1 Challenges

B2B integration involves the integration of different transaction standards with different data representations. These differences lead to the notion of data heterogeneity. Heterogeneity occurs when there is a disagreement about the meaning, interpretation, or intended use of the same or related data. While with distributed database systems four types of information heterogeneity may arise (i.e., system heterogeneity, syntactic heterogeneity, structural or schematic heterogeneity, and semantic heterogeneity), the integration of external transactions and internal data representations suffer only from structural and semantic heterogeneity. The problem of syntactic heterogeneity does not occur since, as seen in our scenario, both external and internal representations rely on XML for data serialization.

Since we have selected both languages to be represented in XML (the external data model is represented with XML and the internal domain model is represented with XML/OWL), there is no ambiguity on how the languages should be parsed since the grammar of both is the same. Nonetheless, a different view on syntactic can be taken. Since the OWL ontology language is based on graphs as the underlying data model and XML is in essence a tree data model, the syntactic mismatch of the underlying data models can occur as more than one XML representation of an OWL graph can exist. In our work, we take the former view on syntactic heterogeneity.

System heterogeneity has long been solved with interoperable operating systems and infrastructures. As for structural and semantic heterogeneity, the following elements need to be considered in B2B integration:

- Structural heterogeneity. External and internal data representations store their terms and expressions in different document layouts and formats, data models, data structures and schemas [40].
- Semantic heterogeneity. External and internal data representations can be expressed semantically in different ways leading to heterogeneity. Semantic heterogeneity considers the content of representations and its intended meaning [42].

Semantic heterogeneity has been identified as one of the most important and pressing challenges in information sharing [42]. Since schema structures and the meaning of XML schema is defined by programmers or data designers, it becomes difficult to automatically or semi-automatically achieve the structural and semantic integration of data among organizations participating in B2B transactions.

Approaches to the problems of structural and semantic heterogeneity should equip B2B platforms aggregating business networks with heterogeneous, autonomous, and distributed systems with the ability to share and exchange business transactions in a semantically consistent way. Therefore, the following problems need to be tackled:

i) Trading partners have to deal with several transaction standards at the same time. To ensure a correct understanding and interpretation, mappings are required which leads in great expenses for organizations [30].
ii) B2B integration through programming does not scale, because of the high complexity of the interactions of B2B standards [7].
iii) The problem of semantic heterogeneity still applies when data is exchanged using XML according to standard specifications for transactions [6].

These three challenges need to be addressed to create better B2B integration platforms. Section 4 will present the B2BISS system and will explain how each of these points are addressed.

While in our research we create mappings between two different data representation languages with a distinct level of expressiveness to enable the B2B integration of organizations, the same type of challenges occurs when two or more information systems use different ontologies and need to "talk" to each other. In such a scenario, there is also the need to find and represent the correspondences between concepts in these ontologies. For example, given two ontologies, mapping one ontology with another means that for each concept in one ontology, we establish a mapping to find a corresponding concept in the other ontology which has the same or similar semantics (and vice verse). The process of finding correspondences is called ontology mapping, ontology matching, or ontology alignment [25] and usually relies on defining a distance measure between entities [16]. The work presented in this paper can be extended to include the notion of distance measures, as described in [16] and [9], to find and suggest to the user which mappings can constitute potential candidates between an XML entity and an OWL concept.

3.2 Approaches to enable integration

When external and internal data representations exist, an important decision for an organization operating in a specific trading industry domain is whether to use a common schema (i.e., a standard) or to use a private custom made internal schema. In other words, the organizations participating in B2B interactions have to adopt one of the following alternatives:

- Agree on a common schema for external and internal data representations.
- Create mappings between external and internal data representations.

The decision on whether to use the first or the second approach depends on various factors which have been well described in [31]. Some of the more important decision factors for our research scenario include the cost, reliability, adequacy, and competitive advantage that a new system or approach can bring to an organization:

- *Cost.* Since a common schema can be sold and distributed to thousands of organizations it means that the price per unit is significantly lower when compared to developing a private internal schema.
- *Reliability.* A common schema with thousands of users has been tested several time and the probability of inconsistencies or errors is relatively low. The same cannot be said with respect to private custom made schema.
- *Adequacy.* A common schema typically has more features that a custom schema since it is designed to satisfy a large base of customers. However, numerous functions are not important for a particular private business and it is not likely to acquire a common schema that fully satisfies all business requirements.
- *Competitive advantage.* By having a custom made schema with specific key features, an organization can gain advantage over competitors. This will not easily happen if all the industry players are using the same common schema.

Our work is directed at organizations that do not feel comfortable in agreeing to structure their internal data representations based on a particular external representation for communication since this alternative restricts their ability to innovate and differentiate from the competition. Therefore, B2BISS is directed at organizations that favor the creation of private schema to represent their internal data. We are not stating that, in the future, the notion of *ontological commitment* reflecting the agreement among business experts and industries will not happen, we are putting forward an intermediate stage where external and internal data representations coexist.

3.3 One solution: heterogeneous mappings

One conceptual solution that allows organizations to participate in B2B integration using external syntactic XML-based transactions while structuring internal data representations semantically using OWL relies on the use of heterogeneous mappings. An *heterogeneous mapping* is defined as a mapping that transforms instances of a schema into instances of another schema that has a different level of expressiveness.

When the B2BISS system uses a mapping to place a syntactic entity from an XML transaction into an ontology, the entity is transparently, and by design, associated with a concept in a proper ontological context which has relationships (e.g. transitive, symmetric and inverse relations) with other concepts from the domain of discourse. In this aspect, after the transformation, the entity acquires a higher level of clarity and expressiveness by being associated with an ontological concept.

Once an organization receives a syntactic external transaction it is allowed to create heterogeneous mappings between the elements from the transaction and the concepts of the ontology that describes the domain of discourse of the organization. The organization that receives the transaction can create any number of mappings. The mappings are stored in a local repository and can be reused when new, previously unseen, syntactic transactions arrive. When a new syntactic transaction is received by an organization, the repository is queried to determine if some of the syntactic elements have already been mapped to ontological concepts in the past. If all the syntactic elements have already been mapped previously, then it is possible to achieve an automatic transaction-based integration.

While in our scenario we use a local repository, mappings could also be used and shared by different organizations trading in the same business value chain. Since we use well-established languages and tools (such as XSLT languages and parsers), a central repository of shared mappings accessible to various organizations (as described in [20]) would enable a stronger reuse of mappings across organizations and over time it would increase the adoption of the B2BISS approach.

The reuse of mappings is a very important topic. This is for many reasons. First, it increases the reliability of the quality of the mapping as already tested and proven mappings are reused and not developed all over again. Second, the speed of establishing mappings increases as reusing a mapping means that the work does not have to be done again. This is significant as mappings take a long time to achieve. Thirdly, if a transaction format changes, a mapping has to be changed only once and the change then is applied to all reuse cases automatically, making the change process very efficient. The reuse of mappings has not been explored to the greatest possible extent.

4 The B2BISS System

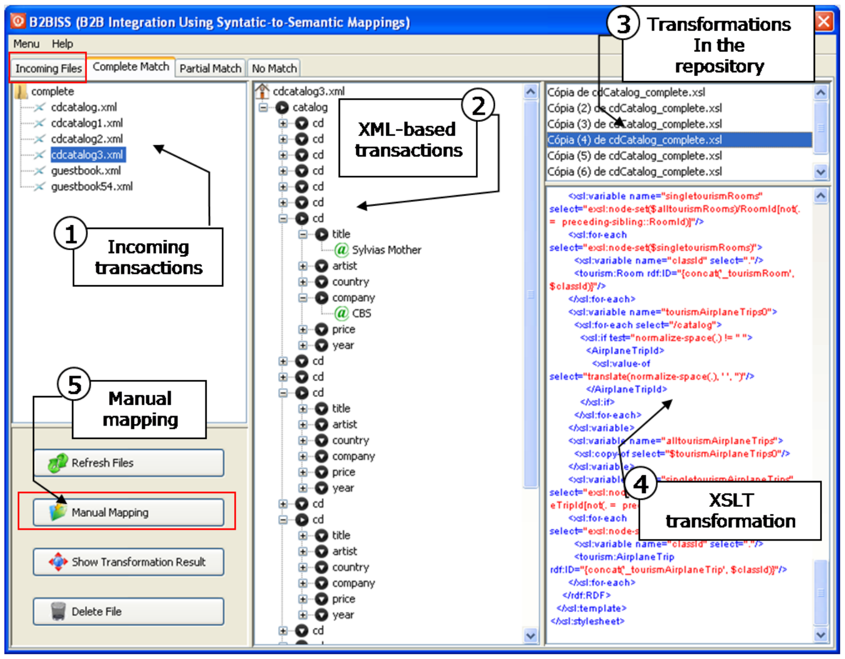
To enable the (semi-)automatic integration of external and internal data representations of organizations participating in B2B networks, we have developed the B2BISS system (B2B Integration using Syntactic-to-Semantic heterogeneous mappings). B2BISS addresses the challenges i), ii) and iii) presented in 3.1 with the management and reuse of mappings which are not programmed but graphically defined by users and encoded with a standard transformation language. Semantic heterogeneity is achieved since the mappings are initially specified graphically by the end user. Later, they can be reused yielding a time and cost reduction in their use and management.

The system uses XSLT [43] language and programs to transform external XML-based transactions to internal OWL instances. The main user interface of the system is shown in Figure 3 and operates in the following way. XML-based standard transactions (defined for example with cXML, CIDX or ebXML) are sent to the organization and are monitored and recorded by the B2BISS monitor (1). When a transaction is received, B2BISS will classify it in one of three categories (see Figure 4): (a) *complete match*, (b) *partial match*, and (c) *no match*.

In case a), B2BISS has found in its repository a mapping or set of mappings that can transform the incoming syntactic transaction to a semantic internal data representation. Figure 3 shows this first case. We can see an XML-based transaction (2) and the set of XSLT transformations (3) that has been found in the repository to transform an XML instance to an OWL instance. One mapping represented with a XSLT transformation is shown (4). In case b), the set of transformations found can only partially transform the XML-based transaction to a semantic representation. In such a case, user intervention will be needed to manually create the missing mappings (5). The application to manually create mappings (called JXML2OWL [41]) is described in Section 4.2. Finally, in case c), no transformations have been found in the B2BISS repository. As a result, new mappings need to be manually created (5). Once they are created they will be stored in the repository for future (re)use.

4.1 Classification algorithm

The algorithm that determines if an incoming syntactic transaction is classified as: (a) *complete match*, (b) *partial match*, or (c) *no match* makes one assumption: an XML-based incoming transaction is completely matched and can be transformed if a set of XSLT transformations (corresponding to a set of mappings) is found for all the XPaths [43] present in the transaction. Having this basis,



```
    <xsl:variable name="singletourismRooms"
select="exsl:node-set($alltourismRooms)/RoomId[not(.
=  preceding-sibling::RoomId)]"/>
    <xsl:for-each
select="exsl:node-set($singletourismRooms)">
      <xsl:variable name="classId" select="."/>
      <tourism:Room rdf:ID="{concat('_tourismRoom',
$classId)}"/>
    </xsl:for-each>
    <xsl:variable name="tourismAirplaneTrips0">
      <xsl:for-each select="/catalog">
        <xsl:if test="normalize-space(.) != " ">
          <AirplaneTripId>
            <xsl:value-of
select="translate(normalize-space(.), ' ', '')"/>
          </AirplaneTripId>
        </xsl:if>
      </xsl:for-each>
    </xsl:variable>
    <xsl:variable name="alltourismAirplaneTrips">
      <xsl:copy-of select="$tourismAirplaneTrips0"/>
    </xsl:variable>
    <xsl:variable          ="singletourismAirplaneTrips"
select="exsl:nod
eTripId[not(. =  pre
    <xsl:for-each
select="exsl:node-s
      <xsl:variable name="classId" select="."/>
      <tourism:AirplaneTrip
rdf:ID="{concat('_tourismAirplaneTrip', $classId)}"/>
    </xsl:for-each>
   </rdf:RDF>
  </xsl:template>
</xsl:stylesheet>
```

Fig. 3. Processing XML-based transactions using B2BISS

we can say that an XML-based transaction (with a set of XPaths S_{xml}) and a set of XSLT transformations (with a set of XPaths S_{xslt}) are a *complete match* when for each XPath in the XML-based transaction there exists an equivalent XPath in a XSLT transformation, i.e., $S_{xml} \subseteq S_{xslt}$. This relation is shown in Figure 4.a). This means that for each XPath there is a XSLT transformation, and therefore, a mapping already exists in the repository and a complete and automatic transformation can be executed.

There is a *partial match* between an XML-based transaction and a set of XSLT transformations if and only if a proper subset of the XML's XPaths exist in the set of XSLT's XPaths, i.e., $S_{xslt} \cap S_{xml} \subset S_{xml}$ (or $S_{xml} \cap S_{xslt} \neq \emptyset$ and $S_{xml} \setminus S_{xslt} \neq \emptyset$). This case is shown in Figure 4.b). When there is a partial match, the user is asked to manually create the missing mappings. Finally, in the case shown in Figure 4.c), an XML-based transaction and a set of XSLT transformations do not have any XPaths in common, i.e., $S_{xml} \cap S_{xslt} = \emptyset$. In this case there is *no match*, and all the required mappings need to be creates manually by the end user. The pseudo code for the classification algorithm is shown in Algorithm 1. The engine that performs the automatic transformations is presented in Section 4.4.

A partial match can be further refined. One idea is to use not only an absolute XSLT path to identify XML nodes but to also use context information. The context information can be represented by taking into account adjacent nodes

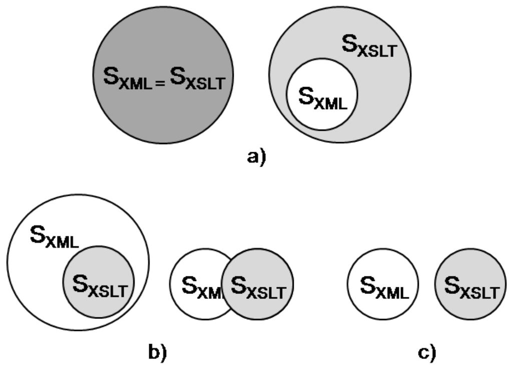


Fig. 4. (a) Complete match, (b) partial match, or (c) no match

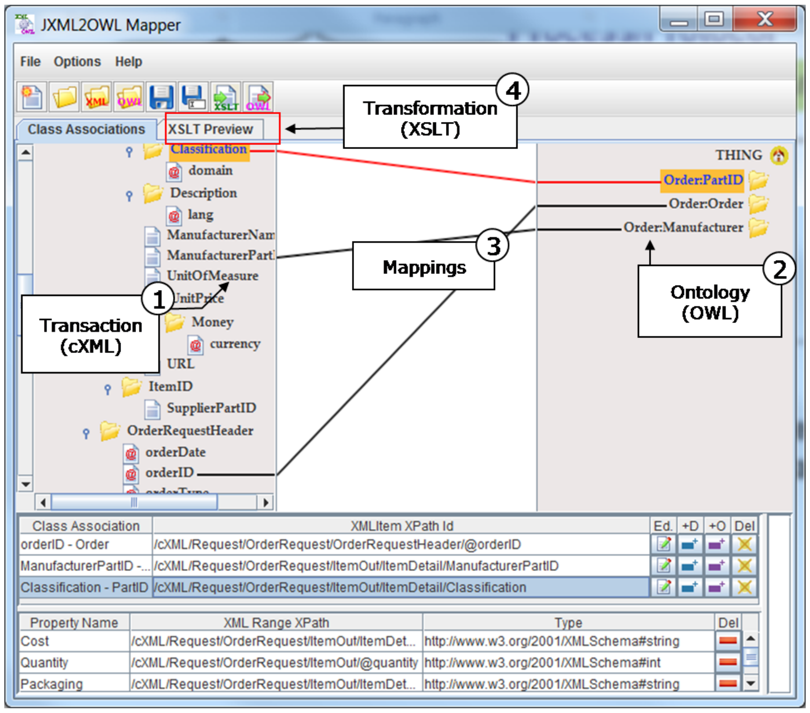
of a target XML node (i.e. a node for which a mapping is needed). Based on this information, the B2BISS system can generate a probabilistic model indicating the estimated validity of a potential mapping. In [40], a large number of alternatives are described to find potential mappings between schemas which can be reused within the B2BISS platform. Examples include internal structure comparison (i.e. accounting for the internal structure of entities, their value range and cardinality) and taxonomical structure (i.e. accounting for the position of the nodes within a taxonomy). These approaches are fundamental to make the discovery of mappings assisted by software applications. Since our work has an emphasis on the transformation of XML entities into OWL concepts, we will not dwell further on this topic.

Algorithm 1 Classification of incoming XML-based transactions

```
if $S_{xml} \subseteq S_{xslt}$ then
    Perform a fully automatic transformation
else
    if $S_{xslt} \cap S_{xml} \subset S_{xml}$ then
        Ask the user to create the missing mappings
    else
        Ask the user to create all mappings
    end if
    Store the new mappings in the repository
    Perform an automatic transformation
end if
```

4.2 Creating heterogeneous mappings with JXML2OWL

When it is necessary to establish a new mapping between an external XML-based transaction and the semantic definition of an OWL-based internal data representation, the user relies on the JXML2OWL (Java XML to OWL) application. The objective of JXML2OWL is to provide a user-friendly, interactive, manual mapping tool. Figure 5 shows the application with the cXML purchase order from Listing 3 on the left side and the ontology from Listing 4 and Figure 2 on the right side of the application.



Class Association	XMLItem XPath Id	Ed.	+D	+O	Del
orderID - Order	/cXML/Request/OrderRequest/OrderRequestHeader/@orderID				
ManufacturerPartID -...	/cXML/Request/OrderRequest/ItemOut/ItemDetail/ManufacturerPartID				
Classification - PartID	/cXML/Request/OrderRequest/ItemOut/ItemDetail/Classification				

Property Name	XML Range XPath	Type	Del
Cost	/cXML/Request/OrderRequest/ItemOut/ItemDet...	http://www.w3.org/2001/XMLSchema#string	
Quantity	/cXML/Request/OrderRequest/ItemOut/@quantity	http://www.w3.org/2001/XMLSchema#int	
Packaging	/cXML/Request/OrderRequest/ItemOut/ItemDet...	http://www.w3.org/2001/XMLSchema#string	

Fig. 5. Creating heterogeneous mappings using JXML2OWL

While XSLT transformations for small transactions can be written quite easily, the complexity and error rate increase dramatically with the size of transactions. Additionally, it is also difficult to maintain and modify existing XSLT programs. Therefore, JXML2OWL is an indispensable application to manage and create heterogeneous mappings and it is divided into two main parts. In the left side of Figure 5, an XML schema of an external transaction is represented (1). In the right side, the OWL schema defined by an internal data representation is shown (2). In between, there is the mapping zone (3), it is possible to drag-and-drop elements between the two schema to easily create heterogeneous mappings. The application allows exporting transformation rules, generated according to the mapping performed, as an XSLT document (4). B2BISS stores well tested and reusable transformation between XML-based transactions and

OWL instances in its repository. These XSLT mappings are then used by B2BISS transformation engine.

JXML2OWL is divided into two sub applications: the JXML2OWL API and the JXML2OWL Mapper. The architecture of the application is shown in Figure 6. The API [4] is a generic and reusable open source library for mapping XML schema to OWL ontologies for the Java platform while the Mapper is an application with a graphical user interface developed in Java Swing that uses the API and eases the management of mappings.

The mapping tool supports mappings between any XML schema (XSD and DTD) to concepts (classes and properties) of any OWL ontology. According to the mapping performed, the tool generates mapping rules as an XSL document that allows the automatic transformation of any XML data, that is, any XML document validating against the mapped schema, into instances of the mapped ontology. Generated mapping rules are wrapped in an XSL document to easily support instance transformation. The XSL document generated by JXML2OWL can be used by any XSLT processor to automatically transform instances of the mapped schema into instances of the ontology. The XSLT choice becomes even more obvious, considering that in our approach OWL was specified with the XML syntax.

It should be stated that XML is just one of the possible syntaxes to represent OWL ontologies. Other increasingly popular data representations include Notation3 (commonly known as N3), Turtle and JSON. They are all non-XML serialization of RDF graphs. N3 was designed to be more compact and readable than XML/RDF. Turtle, a subset of N3, is a more compact and readable alternative. JSON is similar to Turtle, with the advantage of being represented in a language which is easier to parse. Independently of the language chosen to represent organizational internal ontology-based model, XSLT can always be used to perform the transformation of external XML-based documents into internal ontologies.

As illustrated in Figure 6, an JXML2OWL API exists to the Mapper component and the XSLT processor is used by the Instance Generator module.

4.3 The structure of heterogeneous mappings

This section presents the structure to specify mappings between XML and OWL. Several mappings are supported: one-to-one, one-to-many, many-to-one and many-to-many. Therefore, the notation allows mappings from one XML node to several OWL concepts and mappings from several XML nodes to one OWL concept. This section also presents several aspects that must be taken into account when writing an algorithm to perform instance transformation according to created mappings.

The notation The notation to specify mappings between elements of an XML schema and resources defined by an OWL ontology is represented in Table 1.

[4] The API is available at http://jxml2owl.projects.semwebcentral.org/

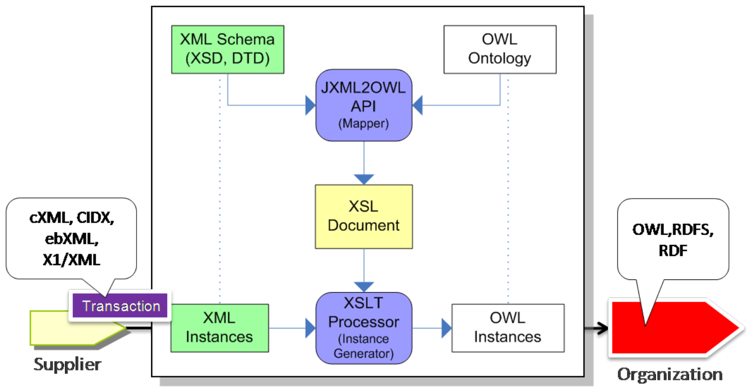


Fig. 6. XML to OWL mapping

Just as with OWL ontologies, which are mainly defined by classes, datatype and object properties, heterogeneous mappings are classified according to three distinct types:

- Class mapping: Maps an XML node to an OWL concept
- Datatype property mapping: Maps an XML node to an OWL datatype property
- Object property mapping: Relates two class mappings to an OWL object property

Table 1. Heterogeneous mapping notation

Mapping	Notation
Class	(OWL Class URI, XPath expression) (OWL Class URI, XPath expression, ID XPath expression)
Datatype Property	(OWL Datatype Property URI, domain class mapping, XPath expression)
Object Property	(OWL Object Property URI, domain class mapping, range class mapping)

OWL resources (classes, object and datatype properties) are addressed using their URI references while XPath expressions are used to address the mapped XML nodes. The use of XPath expressions allows distinguishing several XML nodes with the same name but with different ancestors and permits mapping them to their corresponding OWL concepts. It is also possible to use XPath predicates to enable conditional mappings.

An examination of Table 1 reveals that class mappings are defined by pairs containing the URI reference of the mapped OWL class as well as an XPath expression identifying the mapped XML nodes. Such a pair means that an instance of the mapped OWL class is created for each XML node matching the XPath expression. As an alternative, it is also possible to create class mappings with triplets where the XML node used to compute the IDs of the generated instances are directly specified. Additionally, Table 1 shows that not only property mappings are specified with triplets, but also that class mappings are used to define property mappings. Such a solution enables a complete support of property mappings in the context of many-to-many mappings.

The following example shows how class, data type and object property mappings are created. Let us consider the OWL purchase order ontology from the Hiltrix Corp. from Listing 3 as well as the cXML purchase order transaction from Listing 4.

Eleven mappings were established. Using the notation from Table 1, we have selected the following 6 representative mappings:

– $cm1 = (Order : Part, /cXML/.../ItemDetail/Classification)$
– $cm2 = (Order : Manufacturer, /cXML/.../ManufacturerPartID)$
– $dp1 = (Order : www, cm2, /cXML/.../ItemDetail/URL)$
– $dp2 = (Order : quantity, cm1, /cXML/.../ItemOut/@quantity)$
– $op1 = (Order : hasManufacturer, cm1, cm2)$
– $op2 = (Order : hasPart, cm2, cm1)$

The *cm*1 mapping (*cm*1 stands for class mapping) indicates that an instance of PART class is created for each XML node matching the specified XPath expression, i.e. /CXML/.../ ITEMDETAIL/CLASSIFICATION. Therefore, three instances are created, one for "50202301" (UNSPSC code for water), one for "12352206" (UNSPSC code for tissues), and one for "50202303" (UNSPSC code for frozen juices). The *dp*1 and *dp*2 indicate that for each instance created from the *cm*1 and *cm*2 class mappings, two datatype properties, WWW and QUANTITY, respectively, are also created. Again, it is also necessary to compute the relative path to find the value used to fill the property, which is for the *dp*1 mapping the XPath expression /CXML/.../ITEMDETAIL/URL and for the *dp*2 mapping /CXML/.../ITEMOUT/@QUANTITY. The *op*1 mapping indicates that each instance created from the class mapping *cm*1 is related to an instance created from the class mapping *cm*2 using the relationship HASMANUFACTURER. In other words, a *part* has a *manufacturer*. An inverse relationship also exist and it is named HASPART. The previous six mappings are stored internally in JXML2OWL in XML as illustrated in Listing 5.

4.4 The transformation engine

The mappings created with JXML2OWL are used by the B2BISS Transformation Engine to automatically transform external incoming syntactic transactions into an internal semantic data representation. The engine determines the transaction type and queries the B2BISS repository for a suitable transformation.

Figure 7 illustrates the six mappings previously described: *cm1*, *cm2*, *dp1*, *dp2*, *op1*, and *op2*.

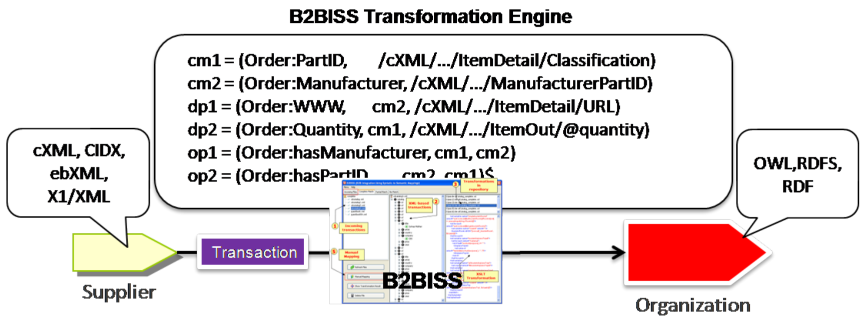

B2BISS Transformation Engine
cm1 = (Order:PartID, /cXML/.../ItemDetail/Classification)
cm2 = (Order:Manufacturer, /cXML/.../ManufacturerPartID)
dp1 = (Order:WWW, cm2, /cXML/.../ItemDetail/URL)
dp2 = (Order:Quantity, cm1, /cXML/.../ItemOut/@quantity)
op1 = (Order:hasManufacturer, cm1, cm2)
op2 = (Order:hasPartID, cm2, cm1)$
cXML, CIDX, ebXML, X1/XML
Supplier
Transaction
B2BISS
OWL,RDFS, RDF
Organization

Fig. 7. B2BISS Transformation Engine

It should be clear that the mappings can only be used to transform XML-based transactions to XML/OWL data representations at this stage. Nevertheless, the engine and mapping application can be extended, if needed, to accommodate other formats used for knowledge representation (for example, N3 and Turtle). In our simple example, the 11 mappings established have generated an XSLT file with 245 instructions. A complete purchase order typically generates 3000 to 4000 instructions. Theses instructions are the basis for the transformation of instance from XML to OWL. The number of XSLT instructions required clearly shows that adequate tool support enables to reduce costs and speedup the integration of heterogeneous schema.

Listing 6 shows the ontology instances that are created when the cXML purchase order from Listing 3 is received and mapped to the Hiltrix Corp. internal purchase order ontology.

The instances of the purchase order ontology capture the products that were ordered by the Aldo Corp. As a result, Hiltrix Corp. has to deliver 10 units of Poland Spring water at a price of $1.20 for each unit, 20 units of Scottex tissues at $1.05, and 30 eight pack of Minute Maid juices at $4.55 per each pack. The relationships (i.e. object properties) that exist between the product instances are described in Listing 4.

4.5 Empirical evaluation

As stated in Section 1, the B2BISS system allows organizations to decrease the time and cost of integration by providing a flexible and easy-to-use graphical tool and a platform for mapping management. Such an infrastructure is instrumental for a rapid and cost-effective B2B integration [37]. To address the *cost and time effectiveness* and *scalability* of the solution we have carried out two experimental evaluations conducted based on two complete case studies to demonstrate empirically the validity of the B2BISS system:

- Converting cXML business transactions to a semantic data model [46].
- Mapping from HR-XML resume to a semantic data model [44].

When compared to creating the mappings by developing software modules, and as expected, we found out that graphically defining mappings was faster, less expensive, required less knowledge, and the process was less error-prone.

Listing 1 shows 10% of the XSLT code which corresponds to the mappings needed to transform automatically XML-based external representations with internal data representations expressed with OWL ontologies of our running scenario. As it can be seen, the graphical creation of mappings from Figure 5 is easier, faster, less expensive and less error prone than creating the mapping manually using XSLT. Stronger advantages were observed when mappings started to be stored in the repository and started to be reused.

Listing 1. Extract (10%) of the XLST mapping file generated

```
<xsl:variable name="singleOrderOrders" select="exsl:node-set
      ($allOrderOrders)/
      OrderId[not(. = preceding-sibling::OrderId)]"/>
 <xsl:for-each select="exsl:node-set($singleOrderOrders)">
  <xsl:variable name="classId" select="."/>
   <Order:Order rdf:ID="{concat('_OrderOrder', $classId)}">
    <xsl:variable name="propshasParts0">
     <xsl:for-each select="$root/cXML/Request/OrderRequest/
                             ItemOut/ItemDetail/Classification">
      <xsl:if test="normalize-space(.) != ''">
       <xsl:variable name="hasPartsProp" select="."/>
        <xsl:for-each select="../../../OrderRequestHeader/
                                                   @orderID">
         <xsl:if test="translate(normalize-space(.),
                                           ' ', '') = $classId">
          <hasPartsId>
           <xsl:value-of select="translate(normalize-space
                                 ($hasPartsProp), ' ', '')"/>
          </hasPartsId>
         </xsl:if>
        </xsl:for-each>
       </xsl:if>
      </xsl:for-each>
     </xsl:variable>
     <xsl:variable name="allPropshasParts">
      <xsl:copy-of select="$propshasParts0"/>
       </xsl:variable>
        <xsl:variable name="singleProphasParts"
```

We have also determined that the solution proposed is scalable from the transformation point-of-view. XML transactions with 385 lines and 10.2 kB of size where transformed into 28 kB OWL ontologies in only 0.266 seconds. XML transactions with 3805 lines and 102 kB took 3.734 seconds to be transferred into an ontology with 254 kB. Since those values (size and number of lines) are typical for external XML data representations for transactions, we can conclude that the platform is scalable for industry domains which use XML-based standards such as cXML and HR-XML. Furthermore, since B2B is usually asynchronous, a variation in performance is not a strong penalty for the approach presented. While it is necessary to write performing code, it will not cause major drawbacks if there are slight variations.

While we recognize that a deeper study on the cost and time effectiveness and scalability of the B2BISS system is necessary, we believe that the two case studies explored and tested constitute a good starting point for a first evaluation.

5 Using B2BISS in an organizational context

In this section we will describe 5 scenarios that illustrate the use of the B2BISS system. The scenarios involve two parties: the Hiltrix Corporation and its customers. Four distinct customers (Aldo Corp., Beplus Corp., Celsius Corp., Degree Corp., and Estar Corp.) will send cXML purchase orders to Hiltrix which will use B2BISS to handle and manage the orders to guarantee that they are transformed into purchase orders stored in its internal simplified ontology. The Hiltrix Corp. has internally developed the ontology described in Listing 4 (in reality, Hiltrix has a larger ontology which aggregates hundreds of concepts that are used daily while interacting with customers). No heterogeneous mappings have been yet defined and, therefore, the B2BISS mapping repository is empty.

5.1 Scenario A. Receiving a purchase order for the first time

In this first scenario, Hiltrix receives a purchase order sent by Aldo Corp. The cXML purchase order is sent to the running B2BISS system. The *Incoming Files* section of the system confirms the receipt of the purchase order. The file is processed to determine if it is a *partial match*, a *complete match* or a *no match* (see section 4.1). The purchase order is moved to the *no match* section of the system shown in Figure 3 since the repository does not contain any heterogeneous mapping yet.

At this point, Hiltrix's managers must create heterogeneous mappings from the purchase order to its internal ontology so they might be able to receive and transform this order, and identical orders, in the future. The managers select the incoming purchase order and choose the *manual mapping* option (see 3). This option launches the JXML2OWL application. Since we are interested in transforming all the data of the purchase order, a complete mapping is established.

Once all heterogeneous mappings are created, the resulting XSLT transformation set produced by the JXML2OWL application (see an example in Listing 5) is automatically imported by the B2BISS application. The purchase order is moved to the *complete match* section and automatically transformed into instances of the Hiltrix ontology (examples of the ontology instances created are illustrated in Listing 6.)

The implementation of the algorithm (Algorithm 1) that determines if the B2BISS system has received an order which is a *partial match*, a *complete match* or a *no match* is straightforward. Since the JXML2OWL application generates XSLT transformation sets with XPaths with a full path format for *for-each* tags, for example,

```
<xsl:for-each
      select="$root/cXML/Request/OrderRequest/ItemOut/@quantity">
```

it is only required to navigate through the XSLT structure and extract the *select* content of the *for-each* tags. Having all the extracted XSLT XPaths in a set, they can be compared to the set of all the XPaths that exist in the cXML purchase order sent by Aldo Corp. The comparison of these two sets determines their degree of match. In this first scenario, the purchase order has been classified initially as a *no match*. Once heterogeneous mappings were created, the purchase was classified as a *complete match* and an automated transformation was carried out.

5.2 Scenario B. Receiving a purchase order for the second time

In the previous scenario, we have received a purchase order for the first time. We performed a set of heterogeneous mappings and the XSLT transformations were imported to the B2BISS system. At this point, the system has one set of XSLT transformations in the B2BISS repository and is able to automatically perform transformations on purchase orders from the Aldo Corp.

Beplus Corp. sends to the Hiltrix Corp. a second purchase order that is identical to the one from the first scenario and that Aldo Corp. has used. The purchase order is added to the *Incoming Files* section and it is processed in order to define if it is a *partial match*, a *complete match* or a *no match*. The purchase order is moved to the *complete match* section because there is a set of XSLT transformations in the repository that can completely transform it. Since there is no need to perform any additional mapping, the transformation is directly applied and instances of the Hiltrix's purchase order ontology are automatically created.

5.3 Scenario C. Receiving a purchase order with new elements

In the previous scenario, Hiltrix Corp. received a purchase order that was automatically transformed upon arrival. In this third scenario, Hiltrix receives a different purchase order from Celsius Corp. that has one new element when compared to the previous orders. This scenario can occur since Celsius may decide to use additional optional fields of the cXML specification. It can also occur that an organization decide to use cXML by making modifications to satisfy business requirements. As a result, Celsius Corp. decided to use the Shipping element from the cXML specification to requested the products (i.e., Water, Tissues, and Orange Juice) to be shipped via FedEX at a cost of $35.50. The new cXML element is illustrated in Listing 2.

Listing 2. The new added cXML Shipping element

```
<!-- The Shipping element is optional. If present, it is
interpreted as a delivery charge. trackingDomain is optional.
If present, it is interpreted as a delivery method. -->

<Shipping trackingDomain="FedEx">
 <!-- The Money element is mandatory. It is the expected value
       of the delivery charge (excluding tax). It must be
       numeric, maximum 2 decimals. currency is mandatory and
       must be the 3 character ISO 4217 code e.g. "GBP" or
       "USD". -->

 <Money currency="USD">35.50</Money>

 <!-- The Description element is optional. If present, it is
       interpreted as description of the charge and any
       delivery instructions.-->

 <Description xml:lang="en-US">
    FedEx for delivery between 9:00 and 17:00
 </Description>

</Shipping>
```

When the purchase order is received, it is classified as *partial match* since there is at least one element that cannot be processed using the existing XSLT transformation set. In other words, the XPath set of the XSLT transformation set stored in the B2BISS repository is a proper subset of the XPath set of the purchase order received (case b) of Figure 4.

At this point, Hiltrix has three options: 1) create a full new set of mappings, 2) edit an existing set of mappings or 3) execute a partial transformation. If it chooses to perform a new set of mappings, it will carry out the steps described in scenario A. The next time Hiltrix receives a purchase order from Celsius Corp. it will carry out the steps from scenario B. Choosing the second option, to edit an existing set of mappings, requires a similar set of actions to be performed but instead of creating a full set of new mappings, only one mappings is added. On the other hand, Hiltrix can also choose to execute a partial transformation (option 3). In such a case, the <Shipping> element from the cXML purchase order will not lead to a new instance in the ontology. Hiltrix decides to create a new set of heterogeneous mappings to satisfy the business requirements of Celsius Corp.

5.4 Scenario D. Receiving a purchase order with fewer elements

In the previous scenario, Hiltrix could have chosen to perform a new set of mappings or to execute a partial transformation. Since Hiltrix opted to create a new set of mappings, at this time, 5 files are stored in the B2BISS system and repository:

1. The purchase order from Aldo Corp. (scenario A)
2. The purchase order from Beplus Corp. (scenario B)
3. The purchase order from Celsius Corp. (scenario C)
4. The XSLT transformation set for purchase orders from Aldo Corp. and Beplus Corp. (scenario A and B)
5. The XSLT transformation set for purchase orders from Celsius Corp. (scenario C)

For this scenario, Hiltrix receives a purchase order from Degree Corp. The order is similar to the purchase order received from Aldo Corp. but Degree Corp. did not use one of its cXML elements (for example, the UNSPSC classification of the products ordered). As a result, the XPath set for this new purchase order is a subset of the XPath sets from the two XSLT transformation sets stored in the B2BISS repository. This situation is described in case a) of Figure 4). When the purchase order from Degree Corp. arrives at the system, it is classified as *complete match* because all the purchase order's XPaths have a XSLT XPath in the transformation sets. The request order can be processed automatically.

5.5 Scenario E. Receiving a purchase order with unknown elements

In this last scenario, Hiltrix receives a purchase order from Estar Corp. for which no XSLT transformation set can be found in the B2BISS repository. In other words, none of the XPath from the request order was found in any of the XSLT transformation sets. This situation is described in case c) of Figure 4. In such a case, Hiltrix has to create a new customized set of mappings that will generate a new XSLT transformation set. The purchase order is automatically classified as *no match*. Once the mappings set for this new cXML order is done, Hiltrix can receive similar purchase orders in the future and accomplish automatically a *complete match* and subsequent transformation.

Based on the scenarios presented, the B2BISS system at Hiltrix Corp. has the following purchase orders and XSLT transformation sets stored in its repository:

1. The purchase order from Aldo Corp. (scenario A)
2. The purchase order from Beplus Corp. (scenario B)
3. The purchase order from Celsius Corp. (scenario C)
4. The purchase order from Degree Corp. (scenario D)
5. The purchase order from Estar Corp. (scenario E)
6. The XSLT transformation set for the purchase order from Aldo Corp. (scenario A, B and D)

7. The XSLT transformation set for the purchase order from Celsius Corp. (scenario C)
8. The XSLT transformation set for the purchase order from Estar Corp. (scenario E)

As more purchase orders are received and additional mapping sets are created, Hiltrix Corp. and its B2BISS system will converge into a stable state were the number of new or unique purchase orders sent for processing which require manual intervention from managers will be marginal. The reuse of mappings across organizations and purchase orders types can dramatically reduce the costs of integration since a typical purchase order can require more than 3000 lines of instructions to transform an cXML request into an OWL-based internal semantic data representation.

6 Integrating B2BISS into SAP PI

This last section explains and describes how the concepts presented in this paper can be technologically integrated into a commercial system. The SAP Process Integration[5] [29] is the enterprise platform supplied as part of SAP NetWeaver solution. It is an integration platform for SAP and non-SAP systems which makes possible the construction of B2B scenarios based on the exchange of business transactions represented with various standards. It allows reducing the cost and development time of integration projects.

At the technical infrastructure, SAP PI uses XML-based transactions in order to connect SAP components with non-SAP components. It supports well-known industry data exchange standards such as cXML, RosettaNet, UCCnet, papiNet, HL7 and PIDX. When an application needs to send or accept a transaction, SAP PI creates a channel to transport the transaction by assuring that it is accepted and delivered exactly according to pre-defined business rules. At the business level, generally an agreement must be reached between the trading partners on how the electronic transactions will be instantiated and transmitted.

Figure 8 shows the architecture overview of SAP PI and how it is used in conjunction with the B2BISS system. The basic infrastructure has a simple structure: a sender application system (left) talks to a central PI Server (center), which sends the XML-based transactions to a receiver application system (right). The integration server is the central part of the infrastructure. It receives transactions and applies routing and mapping rules to these transactions and, finally, sends them to the receiving application. The integration repository provides knowledge available at design time and runtime, such as mappings, interfaces, and components. The information in the integration repository is used by the integration directory, which adds configuration-specific information that is needed for execution such as detailed description of routing rules, active services, executable mappings, etc.

[5] http://www.sap.com/platform/netweaver/components/pi/index.epx

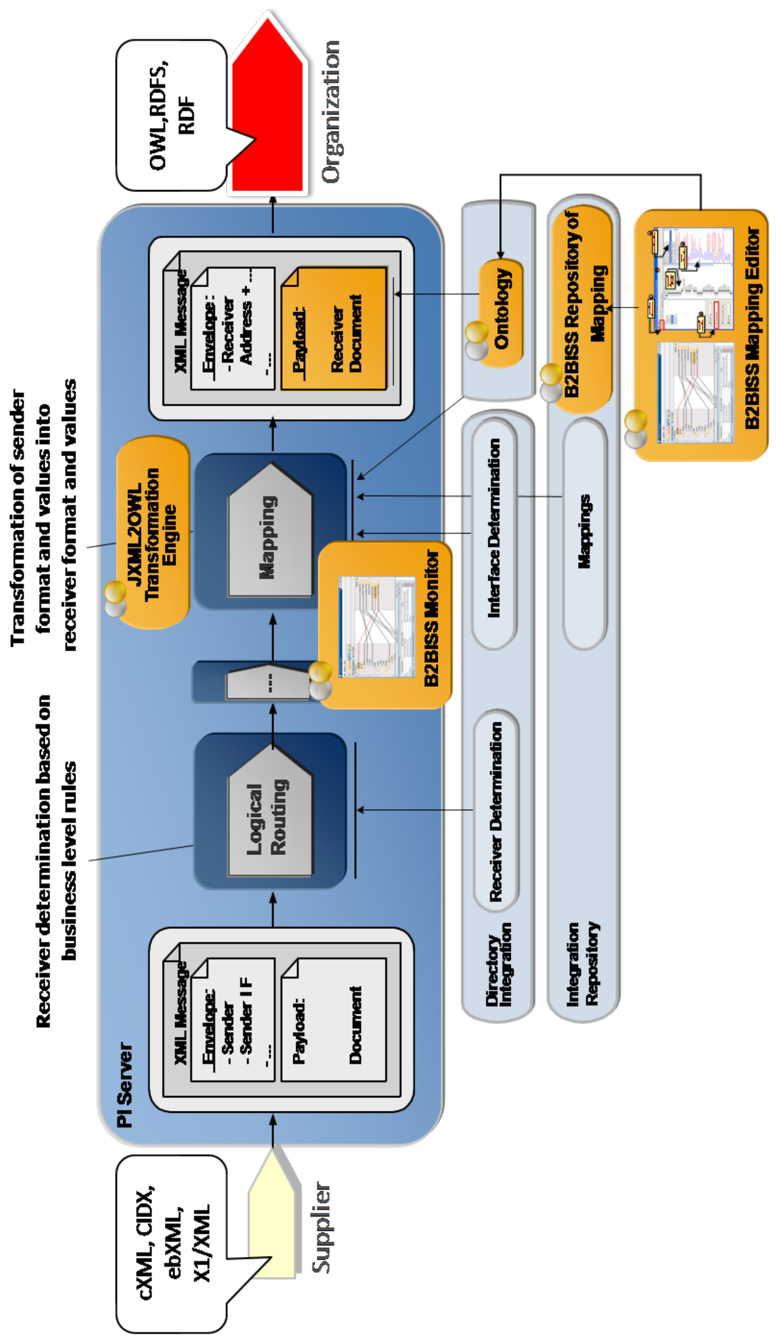


Fig. 8. Extending SAP PI architecture with B2BISS system

Due to its modularity and its architecture, B2BISS can be easily integrated into SAP PI platform. The B2BISS system is integrated into SAP PI routing component. This module will allow the detection of incoming XML-based transactions that contain possibly known, or unknown, elements (i.e., a *complete match*, a *partial match* or *no match* according to the classification Algorithm 1). When such elements are found, transformations to OWL instances are carried out automatically. For these transformations to take place, it is necessary to use the mappings that have been generated by the JXML2OWL application and that have been stored into the SAP PI repository. Additionally, the SAP PI platform needs to be extended with an ontology management system (currently the most popular solution to implement an ontology system is to use a relational database as a cornerstone). When incoming transactions arrive to the platform, the B2BISS monitor can be used to solve scenarios that may arise when only *partial matches* or *no matches* occur. The management of mappings is the responsibility of the B2BISS mapping editor.

The B2BISS system brings several advantages for SAP PI. The approach provides a new set of building blocks that give a considerable step forward in terms of semantic integration. Only one integration infrastructure is needed for B2B integration. The mappings stored are based on XML and open standards (XPath and XSLT). This promotes the ability to provide and reuse this knowledge for partners, customers, third-parties and legacy systems, taking it well beyond a pure SAP-to-SAP integration. As all transactions are routed through one system, organizations need to monitor only SAP PI instead of having to monitor several systems. The ability to modify mappings and add new mappings without impacting the involved systems gives organizations the flexibility to react quickly to dynamic changes in business networks.

7 Related work

The projects described in this section are related to the B2BISS approach presented in this paper to achieve an external (syntactic) to internal (semantic) B2B integration. However, none of them supports the management of heterogeneous mappings and instance transformation from external XML-based standards to an internal representation expressed using an existing domain ontology specified in OWL. The fact that the ontology already exists in B2BISS is relevant since a few related projects have the aim to transform XML schema into new OWL ontologies. The related work can be divided into two main subsections: infrastructures to support B2B integration and tools that enable the mapping between different data representations. These two main subsections correspond to the two interrelated systems that we have presented: B2BISS and JXML2OWL.

Gahleitner and Wolfram [20] focus on the development of a system for managing enterprises' information heterogeneity involving multiple ontologies. The proposed system provides a common interface for reusing ontology mapping information across enterprises' based on the concept of semantic bridges. Compared to our work, B2BISS focus on the management of XML to OWL mappings instead of OWL to OWL mappings. This makes the B2BISS system suitable for an intermediary stage of business ecosystems where some organizations use XML-standards and other organizations have already moved to internal OWL-based ontologies.

Jung et al. [24] describe the design and a prototype implementation of a set of methods for B2B process integration using workflow or business process technology. The design describes a B2B Workflow Reference Model with several interfaces and requirements for workflow integration. The project addresses application integration [38], but not information integration. Application integration is a technology solution where typically most of the product development effort is spend by organizations on issues related to messages, processes, transport and interfaces. The resulting system enables business partners to understand workflow standards and the messaging technologies that can be used. Compared to B2BISS, the work of Jung et al. focuses on the use of business processes for orchestration and coordinating business partners in business networks. Data and semantic integration were not an objective.

WSML, WSMO and WSMX [50] uses semantically annotated services to promote B2B integration by using data and process mediation. Transactions can be semantically enriched using the WSML ontology language. The infrastructure enables domain experts to create mappings between the XML-based standards and WSMO ontologies. The mappings are represented in an abstract ontology mapping language. On the one hand, the use of non-standard languages to establish mappings may be considered a penalty for not allowing its sharing across an industry. On the other hand, the management of mappings is not addressed. This means that for each standard used to represent transactions, a new set of mappings needs to be created. B2BISS solves this problem by using a repository of mappings that can be reused overtime. Incoming XML-based transactions are automatically classified and human intervention is requested by B2BISS when needed. Furthermore, the use of more well-established languages (such as OWL and XSLT) enable a stronger acceptance by the industry.

Díaz, et al. [14] recognizes that information exchanges are often characterized by the need of translating from one data format into another one in order to achieve compatibility between information systems. In their work they present the SIMPLEX system to examine to what extent XML and XSLT support the inter-organizational exchange of business documents. The results were positive. Nevertheless, their approach only targeted the conversion from/to different industry standards represented with XML. Furthermore, they do not advance on the idea of using, reusing, and managing mappings.

A considerable number of tools have the purpose of defining mappings between two distinct data models. Applications which are related to our approach include XML2OWL, FOAM, COMA++, and Lifting XML schema to OWL.

The XML2OWL framework [4] also transforms XML schema (XSD) into a newly created ontology in OWL using XSLT. This project is similar to the JXML2OWL module of the B2BISS system but there are several differences. In fact, this tool creates a new ontology from an XML schema during which the user has no control over the process. Our main objective is different since we allow users to map XML schema to an existing ontology, which is usually richer than the one created by the XML2OWL framework, and appropriately generate an XSLT that automatically transforms instances of the XML schema to instances of the mapped ontology. In this sense, one could argue that XML2OWL is not a mapping tool but a converter.

FOAM (Framework for Ontology Alignment and Mapping) [15] provides a general alignment process based on ontology-encoded semantics focusing on various aspects of ontology alignment. The process uses concepts of features (e.g. labels, structural features, subsumption, restrictions, etc.), similarity, and interpretation to decide if two concepts from distinct ontologies should be aligned. Some of the concepts can be applied to the B2BISS system to automatically make suggestions of possible mappings. Nonetheless, since the alignment and mapping proposed by FOAM are between two ontologies, additional research is needed for the system to propose mappings between XML schema and OWL ontologies.

COMA++ [2] is a schema and ontology matching tool that supports XML schema and OWL ontology documents as data sources and enables a user to identify semantic correspondences between XML schema, between OWL ontologies or even between an XML schema and an OWL ontology. Since COMA++ is a matching tool, it is not really intend to map XML schema to ontologies with the purpose of facilitating the transformation of schema instances into individuals as in B2BISS. Nonetheless, as with FOAM, concepts can be re-used in B2BISS for supporting mapping guidance and to give mapping suggestions to users.

Ferdinand, Zirpins et al. [17] propose an approach to lift an XML schema to a new ontology capturing the implicit semantics available in the structure of XML documents. Their work is divided into two independent parts: automatic mappings from XML to RDF as well as from XML schema to OWL. However, since the mappings are independent, the generated instances may not respect the OWL model created from the XML schema.

Research has also been conducted to convert XML-based B2B standards to ontologies. For example, Kotinurmi and Vitvar [28] added semantics to RosettaNet specifications. Foxvog and Bussler [19] represented the EDI standard in various Semantic Web and logic languages. Our approach does not follow this path, since one requirement is that external XML-based B2B standards and internal ontology-based B2B standards must coexist.

8 Conclusions

Global business environments require an adequate B2B integration for organizations to exchange information with partners, suppliers, and customers. However, if this integration is achieved manually, without any support from automated systems, companies must invest a significant amount of their software maintenance budgets to program, code, and maintain mappings across enterprises. Therefore, tools and applications that can support, ease the integration and reduce transaction costs are fundamental assets for modern businesses.

In a B2B integration scenario, two types of data representations need to be integrated: external and internal. External data representations are used to exchange information with partners. Transaction standards such as cXML and ebXML are XML-based and are widely used in various industries. On the other hand, internal data representations are only visible inside organizations and are used to describe their services, products and purchase orders. While current organizations only rely on database schema, modern approaches involve using semantic languages, such as OWL and RDFS to increase the expressiveness of data. In such a context, one challenge for B2B integration is to find a solution to integrate external XML-based transactions with internal OWL ontology-based data representations.

In this paper, we have described B2BISS, a system that enables the (semi-)automatic integration of external and internal data representations relying on the management of heterogeneous mappings. The solution presented proposes a strategy to map XML-based external representations with internal data representations expressed with OWL ontologies according to a set of mappings managed by B2BISS. Once an organization receives an external transaction it is allowed to create heterogeneous mappings between the elements from the transaction and the concepts of an ontology. The mappings are stored in a local repository and can be reused when new, unseen, transactions arrive. When a new transaction is received, the repository is queried to determine if some of its elements have already been mapped to ontological concepts in the past. If all the elements have already been previously mapped, then it is possible to achieve an automatic transaction-based integration.

The reuse of mappings is important for many reasons. First, it increases the reliability of the quality of the mapping as already tested and proven mappings are reused. Second, the speed of establishing mappings increases as reusing a mapping means that the work does not have to be done again. Thirdly, if an external or internal specification is changed, a mapping has to be changed only once and the change then is applied to all reuse cases automatically, making the change process very efficient.

Our future work will have a particular emphasis on applying the concept of XML to OWL mappings to integrate patient information in the area of healthcare. More precisely, we envision providing surgeons with integrated information about patients in operative situations. Since the XML-based DICOM (Digital Imaging and Communication in Medicine) standard has become widely used in hospitals to ensure the interoperability between devices manufactured by different vendors and, at the same time, medical ontologies have proven to be an important asset in many medical scenarios, we believe that the integration of these two healthcare worlds (syntactic and semantic) can produce important synergies that will support the development of new medical systems and solutions.

9 Acknowledgments

We would like to acknowledge the contributions of João Sobrinho and Daniel Teixeira for programming the B2BISS systems and to Toni Rodrigues and Pedro Rosa for the development of the JXML2OWL mapping tool.

10 Appendix

Listing 3 shows an cXML purchase order [13]. The order specifies the transaction of Poland Spring water, Scottex tissues, and Minute Maid Frozen juice. The amount of the purchase is of $66.80 and it is to be sent to Aldo Corp., a company located in Sunnyvale, California, USA.

Listing 3. cXML transaction to order Water, Tissues and Frozen juices

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
 Ordering three line items for delivery on the same day (a few
 days after the order date).
-->
<!DOCTYPE cXML SYSTEM
          "http://xml.cXML.org/schemas/cXML/1.2.021/cXML.dtd">
<cXML payloadID="32232995@ariba.acme.com"
       timestamp="2000-10-12T18:39:09-08:00" xml:lang="en-US">
 <Header>
  <From>
  <!-- The buying marketplace and member organization. -->
   <Credential domain="AribaNetworkUserId" type="marketplace">
    <Identity>admin@marketplace.org</Identity>
   </Credential>
   <Credential domain="AribaNetworkUserId">
    <Identity>admin@acme.com</Identity>
   </Credential>
  </From>
  <To>
   <Credential domain="DUNS">
    <Identity>942888711</Identity>
   </Credential>
  </To>
  <Sender>
  <!-- This document has passed through Ariba CSN to the
       supplier. -->
   <Credential domain="AribaNetworkUserId">
    <Identity>sysadmin@ariba.com</Identity>
    <SharedSecret>passcode</SharedSecret>
   </Credential>
   <UserAgent>Ariba CSN 33</UserAgent>
  </Sender>
```

```
</Header>
<Request deploymentMode=”production”>
 <OrderRequest>
  <OrderRequestHeader orderID=”DO1234”
          orderDate=”2000-10-12T18:41:29-08:00” type=”new”>
   <Total>
    <Money currency=”USD”>66.80</Money>
   </Total>
   <ShipTo>
    <Address>
     <Name xml:lang=”en”>Aldo Corp.</Name>
     <PostalAddress name=”default”>
      <DeliverTo>Joe Smith</DeliverTo>
      <DeliverTo>Mailstop M-543</DeliverTo>
      <Street>2013 Bloomingdale Street</Street>
      <City>Sunnyvale</City>
      <State>CA</State>
      <PostalCode>90489</PostalCode>
      <Country isoCountryCode=”US”>United States</Country>
     </PostalAddress>
    </Address>
   </ShipTo>
   <BillTo>
    <Address>
     <Name xml:lang=”en”>Aldo Corp.</Name>
     <PostalAddress name=”default”>
      <Street>2013 Bloomingdale Street</Street>
      <City>Sunnyvale</City>
      <State>CA</State>
      <PostalCode>90489</PostalCode>
      <Country isoCountryCode=”US”>United States
      </Country>
     </PostalAddress>
    </Address>
   </BillTo>
   <Payment>
    <PCard number=”1234567890123456”
           expiration=”2001-03-12”/>
   </Payment>
  </OrderRequestHeader>
  <ItemOut quantity=”10”
    requestedDeliveryDate=”2000-10-18” lineNumber=”1”>
   <ItemID>
    <SupplierPartID>1233244</SupplierPartID>
   </ItemID>
   <ItemDetail>
    <UnitPrice>
     <Money currency=”USD”>1.20</Money>
    </UnitPrice>
    <Description xml:lang=”en”>Mineral Water</Description>
```

```
     <UnitOfMeasure>EA</UnitOfMeasure>
     <Classification domain="UNSPSC">
       50202301
     </Classification>
     <ManufacturerPartID>50202301</ManufacturerPartID>
     <ManufacturerName>Poland Spring</ManufacturerName>
     <URL>www.polandspring.com</URL>
    </ItemDetail>
   </ItemOut>
   <ItemOut quantity="20"
      requestedDeliveryDate="2000-10-18" lineNumber="4">
    <ItemID>
     <SupplierPartID>1233245</SupplierPartID>
    </ItemID>
    <ItemDetail>
     <UnitPrice>
      <Money currency="USD">1.05</Money>
     </UnitPrice>
     <Description xml:lang="en">Pocket Tissue</Description>
     <UnitOfMeasure>P8</UnitOfMeasure>
     <Classification domain="UNSPSC">12352206
     </Classification>
     <ManufacturerPartID>12352206</ManufacturerPartID>
     <ManufacturerName>Scottex</ManufacturerName>
     <URL>www.scottex.com</URL>
    </ItemDetail>
   </ItemOut>
   <ItemOut quantity="30"
      requestedDeliveryDate="2000-10-18" lineNumber="5">
    <ItemID>
     <SupplierPartID>1233246</SupplierPartID>
    </ItemID>
    <ItemDetail>
     <UnitPrice>
      <Money currency="USD">4.55</Money>
     </UnitPrice>
     <Description xml:lang="en">Orange Jus</Description>
     <UnitOfMeasure>EA</UnitOfMeasure>
     <Classification domain="UNSPSC">
       50202303
     </Classification>
     <ManufacturerPartID>50202303</ManufacturerPartID>
     <ManufacturerName>Minute Maid</ManufacturerName>
     <URL>www.minutemaid.com</URL>
    </ItemDetail>
   </ItemOut>
  </OrderRequest>
 </Request>
</cXML>
```

Listing 4 shows a simplified version of the OWL ontology used to represent purchasing orders received by the Hiltrix Corporation. The ontology has three main classes: Order, Part and Manufacturer.

Listing 4. The Hitrix Corp. internal purchasing order OWL ontology

```
...
<?xml version="1.0"?>
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:owl="http://www.w3.org/2002/07/owl#">
  <owl:Ontology rdf:about=""/>

  <!-- The class Order -->
  <owl:Class rdf:ID="Order">
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:maxCardinality rdf:datatype="&xsd;int">
          120
        </owl:maxCardinality>
        <owl:onProperty>
          <owl:ObjectProperty rdf:ID="hasItem"/>
        </owl:onProperty>
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:minCardinality rdf:datatype="&xsd;int">
          1
        </owl:minCardinality>
        <owl:onProperty>
          <owl:ObjectProperty rdf:ID="hasItem"/>
        </owl:onProperty>
      </owl:Restriction>
    </rdfs:subClassOf>
  </owl:Class>

  <!-- The class Manufacturer and Part -->
  <owl:Class rdf:ID="Manufacturer"/>
  <owl:Class rdf:ID="Part"/>

  <!-- The data property classification -->
  <owl:DatatypeProperty rdf:about="#classification">
    <rdfs:domain>
      <owl:Class>
        <owl:unionOf rdf:parseType="Collection">
          <owl:Class rdf:about="#Part"/>
          <owl:Class rdf:about="#Manufacturer"/>
        </owl:unionOf>
    </owl:Class>
  </rdfs:domain>
  <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>

<!-- The data property type -->
<owl:DatatypeProperty rdf:ID="type">
  <rdfs:domain rdf:resource="#Order"/>
  <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>

<!-- The data property cost -->
<owl:DatatypeProperty rdf:ID="cost">
  <rdfs:domain rdf:resource="#Part"/>
  <rdfs:range rdf:resource="&xsd;decimal"/>
</owl:DatatypeProperty>

<!-- The data property quantity -->
<owl:DatatypeProperty rdf:ID="quantity">
  <rdfs:domain rdf:resource="#Part"/>
  <rdfs:range rdf:resource="&xsd;int"/>
</owl:DatatypeProperty>

<!-- The data property packaging -->
<owl:DatatypeProperty rdf:ID="packaging">
  <rdfs:domain rdf:resource="#Part"/>
  <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>

<!-- The data property www -->
<owl:DatatypeProperty rdf:ID="www">
  <rdfs:domain rdf:resource="#Manufacturer"/>
  <rdfs:range rdf:resource="&xsd;anyURI"/>
</owl:DatatypeProperty>

<!-- The object property hasManufacturer -->
<owl:ObjectProperty rdf:ID="hasManufacturer">
  <rdfs:range rdf:resource="#Manufacturer"/>
  <rdfs:domain rdf:resource="#Part"/>
  <owl:inverseOf rdf:resource="#hasPart"/>
</owl:ObjectProperty>

<!-- The object property hasPart -->
<owl:ObjectProperty rdf:ID="hasPart">
  <rdfs:range rdf:resource="#Part"/>
  <rdfs:domain rdf:resource="#Manufacturer"/>
  <owl:inverseOf rdf:resource="#hasManufacturer"/>
</owl:ObjectProperty>

<!-- The object property hasItem -->
```

```
  <owl:ObjectProperty rdf:ID="hasItem">
    <rdfs:range rdf:resource="#Part"/>
    <rdfs:domain rdf:resource="#Order"/>
  </owl:ObjectProperty>

</rdf:RDF>
```

Listing 5 shows the six mappings that were stored internally in JXML2OWL in XML as described in 4.2.

Listing 5. Mappings established between the cXML purchase order and the Hitrix Corp. purchase order ontology

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<mapping>
 <xmlURI>http://.../cXML-OR-Patagonia-Corp.xml</xmlURI>
 <owlURI>http://.../Hitrix-Corp-InternalOrder.owl</owlURI>
 <owlPrefix>Order</owlPrefix>
 <associations>
   <class>
    <owlClassName>#Manufacturer</owlClassName>
    <elementXPath>/cXML/.../ManufacturerPartID</elementXPath>
    <IdXPath>/cXML/.../ManufacturerPartID</IdXPath>
   </class>
   <class>
    <owlClassName>#Part</owlClassName>
    <elementXPath>/cXML/.../Classification</elementXPath>
    <IdXPath>/cXML/.../ItemDetail/Classification</IdXPath>
   </class>
   ...
   <datatypeProperty>
    <owlPropertyName>#www</owlPropertyName>
    <owlDomainClassName>#Manufacturer</owlDomainClassName>
    <domainXPath>/cXML/.../ManufacturerPartID</domainXPath>
    <valueXPath>/cXML/.../ItemOut/ItemDetail/URL</valueXPath>
   </datatypeProperty>
   <datatypeProperty>
    <owlPropertyName>#quantity</owlPropertyName>
    <owlDomainClassName>#Part</owlDomainClassName>
    <domainXPath>/cXML/.../Classification</domainXPath>
    <valueXPath>/cXML/..../ItemOut/@quantity</valueXPath>
   </datatypeProperty>
   ...
   <objectProperty>
    <owlPropertyName>#hasPart</owlPropertyName>
    <owlDomainClassName>#Manufacturer</owlDomainClassName>
    <domainXPath>/cXML/.../ManufacturerPartID</domainXPath>
    <owlRangeClassName>#Part</owlRangeClassName>
    <rangeXPath>/cXML/.../Classification</rangeXPath>
   </objectProperty>
```

```
    <objectProperty>
     <owlPropertyName>#hasManufacturer</owlPropertyName>
     <owlDomainClassName>#Part</owlDomainClassName>
     <domainXPath>/cXML/..../Classification</domainXPath>
     <owlRangeClassName>#Manufacturer</owlRangeClassName>
     <rangeXPath>/cXML/.../ManufacturerPartID</rangeXPath>
    </objectProperty>
     </associations>
</mapping>
```

Listing 6 shows the instances that are created when the order from Listing 3 is received by Hiltrix Corp.

Listing 6. Hiltrix Corp. internal purchase order ontology instances

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
   xmlns:owl="http://www.w3.org/2002/07/owl#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   xmlns:Order="http://.../Hitrix-Corp-InternalOrder.owl#">
  <owl:Ontology rdf:about="">
   <owl:imports
     rdf:resource="http://.../Hitrix-Corp-InternalOrder.owl"/>
  </owl:Ontology>
  <Order:Order rdf:ID="_OrderOrderDO1234">
   <Order:hasItem rdf:resource="#_OrderPart50202301"/>
   <Order:hasItem rdf:resource="#_OrderPart12352206"/>
   <Order:hasItem rdf:resource="#_OrderPart50202303"/>
  </Order:Order>
  <Order:Manufacturer rdf:ID="_OrderManufacturer50202301">
   <Order:hasPart rdf:resource="#_OrderPart50202301"/>
   <Order:classification rdf:datatype="&xsd;string">
     Poland Spring
   </Order:classification>
   <Order:www rdf:datatype="&xsd;string">
     www.polandspring.com
   </Order:www>
  </Order:Manufacturer>
  <Order:Manufacturer rdf:ID="_OrderManufacturer12352206">
   <Order:hasPart rdf:resource="#_OrderPart12352206"/>
   <Order:classification rdf:datatype="&xsd;string">Scottex
   </Order:classification>
   <Order:www rdf:datatype="&xsd;string">
     www.scottex.com
   </Order:www>
  </Order:Manufacturer>
  <Order:Manufacturer rdf:ID="_OrderManufacturer50202303">
   <Order:hasPart rdf:resource="#_OrderPart50202303"/>
   <Order:classification rdf:datatype="&xsd;string">
```

```
      Minute Maid
    </Order:classification>
    <Order:www rdf:datatype="&xsd;string">
      www.minutemaid.com
    </Order:www>
  </Order:Manufacturer>
  <Order:Part rdf:ID="_OrderPart50202301">
    <Order:hasManufacturer
            rdf:resource="#_OrderManufacturer50202301"/>
    <Order:cost rdf:datatype="&xsd;decimal">1.20</Order:cost>
    <Order:quantity rdf:datatype="&xsd;int">10</Order:quantity>
    <Order:packaging rdf:datatype="&xsd;string">
      EA
    </Order:packaging>
  </Order:Part>
  <Order:Part rdf:ID="_OrderPart12352206">
    <Order:hasManufacturer
            rdf:resource="#_OrderManufacturer12352206"/>
    <Order:cost rdf:datatype="&xsd;decimal">1.05</Order:cost>
    <Order:quantity rdf:datatype="&xsd;int">20</Order:quantity>
    <Order:packaging rdf:datatype="&xsd;string">
      P8
    </Order:packaging>
  </Order:Part>
  <Order:Part rdf:ID="_OrderPart50202303">
    <Order:hasManufacturer
            rdf:resource="#_OrderManufacturer50202303"/>
    <Order:cost rdf:datatype="&xsd;decimal">4.55</Order:cost>
    <Order:quantity rdf:datatype="&xsd;int">30</Order:quantity>
    <Order:packaging rdf:datatype="&xsd;string">
      EA
    </Order:packaging>
  </Order:Part>
</rdf:RDF>
```

Bibliography

[1] Dean Allemang and James Hendler. *Semantic Web for the Working Ontologist: Effective Modeling in RDFS and OWL.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2011.

[2] David Aumueller, Hong-Hai Do, Sabine Massmann, and Erhard Rahm. Schema and ontology matching with COMA++. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 906–908, New York, NY, USA, 2005. ACM.

[3] Christopher Baker and Kei-Hoi Cheung, editors. *The Semantic Web - Revolutionizing Knowledge Discovery in the Life Sciences.* Springer-Verlag, Heidelberg, 2007.

[4] Hannes Bohring and Sören Auer. Mapping XML to OWL ontologies. In *Leipziger Informatik-Tage*, volume 72 of *LNI*, pages 147–156. GI, 2005.

[5] Raluca Bunduchi. Trust, power and transaction costs in B2B exchanges – a socio-economic approach. *Industrial Marketing Management*, 37(5):610 – 622, 2008.

[6] Christoph Bussler. B2B protocol standards and their role in semantic B2B integration engines. *Bulletin of the Technical Committee on Data Engineering*, 24(1), 2001.

[7] Christoph Bussler. Modeling and executing semantic B2B integration. In *RIDE '02: Proceedings of the 12th International Workshop on Research Issues in Data Engineering: Engineering E-Commerce/E-Business Systems (RIDE'02)*, page 69, Washington, DC, USA, 2002. IEEE Computer Society.

[8] Christoph Bussler. *B2B Integration: Concepts and Architecture.* Springer, 2010.

[9] Jorge Cardoso. Discovering semantic web services with and without a common ontology commitment. In *The 3rd International Workshop on Semantic and Dynamic Web Processes (SDWP 2006), In conjunction with the 2005 IEEE International Conference Web Services (ICWS 2006)*, pages 183–190, Chicago, USA, 2006. IEEE Computer.

[10] Jorge Cardoso. The semantic web vision: Where are we? *Intelligent Systems, IEEE*, 22(5):84 –88, sept.-oct. 2007.

[11] Jorge Cardoso, Wil van der Aalst, Christoph Bussler, Amit Sheth, and Kurt Sandkuhl. *Inter-Enterprise System and Application Integration: A Reality Check*, pages 3–15. LNBI, Springer, 2009.

[12] Jorge Cardoso and Lytras Miltiadis. *Semantic Web Engineering in the Knowledge Society.* Idea Group, Hershey, PA, 2008.

[13] cXML. cXML user's guide. cXML.org, 1.2.021 november 2009 edition, 2009.

[14] Luis Martín Díaz, Erik Wüstner, and Peter Buxmann. Inter-organizational document exchange: facing the conversion problem with XML. In *SAC '02: Proceedings of the 2002 ACM symposium on Applied computing*, pages 1043–1047, New York, NY, USA, 2002. ACM.

[15] Marc Ehrig and York Sure. FOAM – framework for ontology alignment and mapping; results of the ontology alignment initiative. In Benjamin Ashpole, Marc Ehrig, Jérôme Euzenat, and Heiner Stuckenschmidt, editors, *Proceedings of the Workshop on Integrating Ontologies*, volume 156, pages 72–76. CEUR-WS.org, October 2005.

[16] Jérôme Euzenat and Petko Valtchev. Similarity-based ontology alignment in OWL-Lite. In *European Conference on Artificial Intelligence ECAI-04*, Valencia, España, 2004.

[17] Matthias Ferdinand, Christian Zirpins, and D. Trastour. Lifting XML Schema to OWL. In Nora Koch, Piero Fraternali, and Martin Wirsing, editors, *Web Engineering - 4th International Conference, ICWE 2004, Munich, Germany, July 26-30, 2004, Proceedings*, pages 354–358. Springer Heidelberg, 2004.

[18] Agata Filipowska, Martin Hepp, Monika Kaczmarek, and Ivan Markovic. Organisational ontology framework for semantic business process management. In *BIS*, pages 1–12, 2009.

[19] Douglas Foxvog and Christoph Bussler. Ontologizing EDI semantics. In *ER Workshops*, pages 301–311, 2006.

[20] Eva Gahleitner and Wolfram Wöß. Enabling distribution and reuse of ontology mapping information for semantically enriched communication services. In *Proceedings of the Database and Expert Systems Applications, 15th International Workshop*, pages 116–121, Washington, DC, USA, 2004. IEEE Computer Society.

[21] Yolanda Gil and Varun Ratnakar. A comparison of (semantic) markup languages. In *Proceedings of the Fifteenth International Florida Artificial Intelligence Research Society Conference*, pages 413–418. AAAI Press, 2002.

[22] Martin Hepp, Joerg Leukel, and Volker Schmitz. A quantitative analysis of eClass, UNSPSC, eOTD, and RNTD: Content, coverage, and maintenance. In *ICEBE 05- Proceedings of the IEEE International Conference on e-Business Engineering*, pages 572–581, Washington, DC, USA, 2005. IEEE Computer Society.

[23] Zhang Juan, Hu Ning, and Chen Jianhua. Research of multinational corporations based on information technology. In *E -Business and E -Government (ICEE), 2011 International Conference on*, pages 1 –4, may 2011.

[24] Jae-Yoon Jung, Hoontae Kim, and Suk-Ho Kang. Standards-based approaches to B2B workflow integration. *Computers & Industrial Engineering*, 51(2):321–334, 2006. Special Issue: Logistics and Supply Chain Management.

[25] Yannis Kalfoglou and Marco Schorlemmer. Ontology mapping: the state of the art. *The Knowledge Engineering Review*, 18(01):1–31, 2003.

[26] Won Kim and Jungyun Seo. Classifying schematic and data heterogeneity in multidatabase systems. *Computer*, 24(12):12–18, 1991.

[27] M. Klein, D. Fensel, F. van Harmelen, and I. Horrocks. The relation between ontologies and xml schemas. *Electronic Trans. on Artificial Intelligence*, 2001. Special Issue on the 1st International Workshop “Semantic Web: Models, Architectures and Management”.

[28] Paavo Kotinurmi and Tomas Vitvar. Adding semantics to rosettanet specifications. In *WWW Conference*, pages 1059–1060, 2006.

[29] Mandy Krimme and Joachim Orb. *SAP NetWeaver Process Integration.* SAP PRESS, 2nd edition, 2010.

[30] Fenareti Lampathaki, Spiros Mouzakitis, George Gionis, Yannis Charalabidis, and Dimitris Askounis. Business to business interoperability: A current review of XML data integration standards. *Comput. Stand. Interfaces*, 31(6):1045–1055, 2009.

[31] Ken Laudon and Jane Laudon. *Management Information Systems.* Prentice Hall, Upper Saddle River, N.J., 11th edition, 2009.

[32] Ilkka Melleri, Kari Hiekkanen, and Juha Mykkanen. Service-oriented business to business integration: A systematic literature analysis. In *Service-Oriented Computing and Applications (SOCA), 2010 IEEE International Conference on*, pages 1–6, December 2010.

[33] Juha-Miikka Nurmilaakso, Paavo Kotinurmi, and Hannu Laesvuori. XML-based e-business frameworks and standardization. *Computer Standards & Interfaces*, 28(5):585 – 599, 2006.

[34] Hervé Panetto and Arturo Molina. Enterprise integration and interoperability in manufacturing systems: Trends and issues. *Comput. Ind.*, 59(7):641–646, 2008.

[35] Cinzia Parolini. *The Value Net: A Tool for Competitive Strategy.* John Wiley & Sons, Heidelberg, 1999.

[36] Joe Peppard and Anna Rylander. From value chain to value network: Insights for mobile operators. *European Management Journal*, 24(2-3):128 – 141, 2006.

[37] Giacomo Piccinelli, Anthony Finkelstein, and Tommaso Costa. Flexible B2B processes: the answer is in the nodes. *Information and Software Technology*, 45(15):1061 – 1063, 2003. Special Issue on Modelling Organisational Processes.

[38] Jeffrey Pollock. The big issue: Interoperability vs. integration, 2001.

[39] Michael Porter. *Competitive Advantage: Creating and Sustaining Superior Performance.* Free Press, New York, 1985.

[40] Erhard Rahm and Philip A. Bernstein. A survey of approaches to automatic schema matching. *The VLDB Journal*, 10:334–350, December 2001.

[41] Toni Rodrigues, Pedro Rosa, and Jorge Cardoso. Moving from syntactic to semantic organizations using JXML2OWL. *Computers in Industry*, 59(8):808–819, 2008.

[42] Neal G. Shaw, Ahmad Mian, and Surya B. Yadav. A comprehensive agent-based architecture for intelligent information retrieval in a distributed heterogeneous environment. *Decis. Support Syst.*, 32(4):401–415, 2002.

[43] Aaron Skonnard and Martin Gudgin. *Essential XML Quick Reference: A Programmer's Reference to XML, XPath, XSLT, XML Schema, SOAP, and More.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2001.

[44] João Sobrinho. Mapping from HR-XML resume to a semantic data model. Technical report, Department de Matemática e Engenharias, Universidade da Madeira, Funchal, Portugal, 2007. pp. 25.

[45] Don Tapscott, David Ticoll, and Alex Lowy. Digital capital: harnessing the power of business webs. *Ubiquity*, 2000(May):3, 2000.

[46] Daniel Teixeira. Converting cXML business transactions to a semantic data model. Technical report, Department de Matemática e Engenharias, Universidade da Madeira, Funchal, Portugal, 2007. pp. 19.

[47] Efraim Turban, Jae K. Lee, David King, Ting Peng Liang, and Deborrah Turban. *Electronic Commerce 2010*. Pearson Higher Education, 2010.

[48] Marko Vujasinovic, Nenad Ivezic, Boonserm Kulvatunyou, Edward Barkmeyer, Michele Missikoff, Francesco Taglino, Zoran Marjanovic, and Igor Miletic. Semantic mediation for standard-based b2b interoperability. *Internet Computing, IEEE*, 14(1):52 –63, jan.-feb. 2010.

[49] Yong-qiang Yang, Mu-hong Dai, and Hao Chen. An automatic semantic extraction algorithm for xml document. In *Machine Vision and Human-Machine Interface (MVHI), 2010 International Conference on*, pages 41 –44, april 2010.

[50] Maciej Zaremba, Maximilian Herold, Raluca Zaharia, and Tomas Vitvar. Data and process mediation support for B2B integration. In Raul Garcia-Castro, Asunción Gómez-Pérez, Charles J. Petrie, Emanuele Della Valle, Ulrich Kster, Michal Zaremba, and M. Omair Shafiq, editors, *EON*, volume 359 of *CEUR Workshop Proceedings*, 2008.