An Address is not an Address - Specifying Views into B2B Documents

Christian Huemer
Institute for Computer Science and Business Informatics
University of Vienna
Liebiggasse 4/3, 1010 Vienna, Austria
christian.huemer@univie.ac.at

Abstract

Business-to-Business (B2B) electronic commerce has been dominated for a long time by electronic data interchange (EDI) standards, like UN/EDIFACT or X12. These standards are said to be costly to implement and failed in the acceptance of small and medium enterprises (SMEs). XML-based e-business vocabularies seem to provide an alternative. However, we show that traditional EDI standards and XML-based vocabularies have the same problem in common: they require bilateral agreements between business partners in addition to a "commonly shared" document structure. These additional agreements make B2B expensive to implement. It is our goal to support the automation of the agreement process itself. We refer to the concepts of ebXML to specify the choreography of inter-organizational e-business transactions. The proposed extension suggests a way to define unambiguous documents supporting ebXML activities. It is based on specifying views into common business objects that build a document. We use a variant of UML's class, collaboration and activity diagrams for defining these views.

1. Introduction

The most common way to categorize e-commerce is by partner types - which are businesses (B), end-consumers (C) - participating in the commercial transaction. Accordingly, B2C and B2B are the main types of e-commerce transactions. Another partner type might be administration bodies (A). However, this extension is not important for this paper and B will include any kind of organization, be it business or administrative. This paper concentrates on B2B e-commerce, which is characterized by high business value, long-term relationships, complex business processes, intercomputer communications, security, and a multitude of transaction models [11]. The key to successful e-business is application-to-application support of e-business transactions. Application-to-application requires each application to understand informations sent by the other one and, furthermore, to react on business events occurring in a complex business transaction. Once a inexpensive solution for this problem is developed, ad-hoc B2B transactions might be possible. This would eliminate the need for long-term relationships with a high business value.

Application-to-application approaches are nothing new. Since the late 1960s they have been implemented by using the concepts of electronic data interchange (EDI). EDI is defined as the electronic transfer from application-to-application of commercial or administrative transactions using an agreed-upon standard to structure the transaction data. EDI is independent of the transport medium, hardware and software used by the involved business partners (cf. [1] [10] [12]). The field of EDI has been dominated by traditional EDI standards, like X12 or UN/EDIFACT. These standards cover a set of standardized document types (called transaction sets in X12 or message types in UN/EDIFACT). Over the last years many XML-based e-business vocabularies appeared [16]. It is said that XML will get the acceptance by small and medium enterprises (SMEs) traditional EDI standards never had. However, XML is not by itself a solution to any computing problem [6]. If an XML-based solution only focuses on globally valid document structures, the same problems as in traditional EDI will arise.

The approach to traditional EDI standardization implies two major drawbacks. Firstly, it lacks of a choreography of business processes. This means that there needs to be a way to define the business events (and their order) that could eventually happen and an application has to react upon. Secondly, the defined document structures cover a super-set of what is needed by a particular set of business partners. Thus, business partners have to agree both on a choreography of the business processes and on a ruleset to apply each of the document types supporting these processes. Implementing and maintaining these agreements are the reasons why EDI is expensive and only affordable by large corporations.

Agreements on business processes will always be required before business partners can actually do business electronically with each other. However, these agreements need not to be made off-line. This paper's background is automating the agreement process. For defining the choreography of business processes we refer to the relevant specifications of the ebXML initiative [9]. It is our goal to define a technique for specifying business partners' views on documents supporting ebXML transactions. This technique is based on the notation of the Unified Modeling Language (UML) [2]. In future work a representation language for capturing the artifacts of this technique has to be developed. This should allow an

organization to query another organization's view and to automatically map it with its own requirements.

The remainder of this paper is structured as follows: Section 2 illustrates the way UN/EDIFACT (as representative of traditional EDI standards) and xCBL (as representative of XML-based vocabularies) build document types. We elaborate on the weaknesses of these approaches and demonstrate the necessity of additional agreements. Since our approach draws on the ebXML initiative we introduce the main ebXML concepts in Section 4. Section 5 defines by means of a simple example a way of defining exact access rules for B2B documents. These rules make use of UML diagrams and common business objects. We conclude with a short summary.

2. Document types in UN/EDIFACT and xCBL

With the first success stories of XML in the area of EDI [19], one might think that XML will solve all the EDI problems. In this section, we demonstrate that traditional EDI-standards as well as XML-based e-business vocabularies will bear the same problems. For this purpose we have selected UN/EDIFACT as representative of traditional EDI standards and xCBL as representative of XML-based vocabularies. Furthermore, this section should point out what is needed in an agreement on a view of a document structure. This should set the stage for the requirements that have to be met by an electronic support of the agreement process.

2.1. Definition of document types in UN/EDIFACT

UN/EDIFACT defines a standardized structure for each type of business document. Whenever business requirements in any sector or in the case of UN/EDIFACT in any part of the world were discovered, new UN/EDIFACT components were added to the document type structure to capture the business semantics. The result are document type designs that include a lot of optionality without explaining when to use which option. There does not exist any information system capable of processing all the semantics included in a UN/EDIFACT document type. Information systems, which are usually suited to the business requirements of an organization, require only a very small percentage of the document type as input to perform the business transaction. The problem of missing semantics and overloaded document types is handled by EDI branch organizations. They trim down the EDI standard documents to suit the requirements of businesses in a particular sector and/or particular part of the world [20]. An international "standard" document type ends up in multiple different interpretations, so-called message implementation guidelines. Unfortunately, different message implementation guidelines sometimes use different UN/EDIFACT components to capture the same business requirements.

TMWG has analyzed the current EDI standardization approach and high-lighted the main problems, which cause the process to be less efficient than it could be [21]. The main criticism is that there does not exist a well defined method to collect and structure the user requirements. The only output of the standardization process is the UN/EDIFACT structure, which is hard to understand and leaves a lot of space for different interpretations.

To illustrate heterogeneous interpretations we use an extract of a branching diagram of the purchase order document type as an example. The branching diagram is depicted in Figure 1. Among others the following problems are encountered:

- It is not documented which parties are meaningful to be mentioned in a purchase order. The standard allows to express any type of party, for which a coded party qualifier (first data element in Name and Address—NAD) exists. Is it useful to specify the social securities collector's office in a purchase order? Due to the hierarchical design of UN/EDIFACT a definition of a segment is independent of a document type. Accordingly each code for a data element of the corresponding segment is valid in each document the segment appears in.

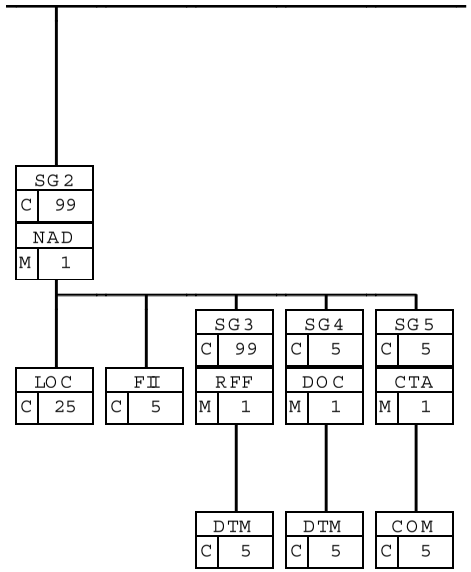

Figure 1 Extract of the Purchase Orders Branching Diagram

- The assignment of multiple addresses to a party is difficult, since the address comes along with the name (or PartyID) in the same segment NAD—establishing a 1:1-relationship. So what to do when assigning multiple addresses to a buyer? A very common opportunity would be to specify another (‘virtual’) party for each address. Another choice - when multiple addresses of the same type are needed - is the ’misuse’ of the location information (LOC). This segment does not allow for textual addresses, but only for coded ones. Most likely, the delivery address is not in an international agreed code list, like LOCODE. So partners use proprietary code lists.
- How is the party identified—by unstructured name and address lines, by structuring name, street, city etc., or just by party identification? The standard’s NAD segment is capable of handling all of them, but is the partner able to process all of them? The standard even covers an element for identifying the list which the party identification refers to. But will the partner have access to that list?
- Is it useful to specify a contact person for all parties or just for specific types of parties. The standard does not include any constraints to be specified according to data values—and party type is an instance of the corresponding qualifier.
- It is possible to refer to documents of a party. First of all, what kind of documents are valid (and for which type of party)? Secondly, what segment should be used—the reference (RFF) or the document (DOC) segment? Both seem to be valid.

According to the problems mentioned above, it is easy to recognize that the definition of a UN/EDIFACT standard document is not semantically complete. Not all existing dependencies and constraints of the business transaction are evident to the user. A UN/EDIFACT standard document is a superset-like document type which is able to handle all situations in the considered business transaction. Document types include a lot of optionality to chose from and the same information can be passed in different ways within a document. Producing an UN/EDIFACT document conforming to the standard does not mean that the receiver of this document supporting UN/EDIFACT is able to process it. Before exchanging documents organizations must bilaterally determine which subset of the document type and rule set their applications are able to support.

2.2. Definition of document types in xCBL 2.0

In order to compare the UN/EDIFACT approach with an XML-based approach we will have a look at the proposed solution for specifying parties in a purchase order in an XML-based solution. As mentioned before, there exists a huge number of XML-based dialects and unfortunately they all differ in their particular way to structure party information. To illustrate our approach we choose the (extensible) Common Business Library 2.0 or xCBL, one of the most popular XML-based approaches. xCBL is a set of XML building blocks and a document framework that allows the creation of robust, reusable XML documents for e-commerce [5]. xCBL includes a comprehensive set of business documents and makes the full document set and component library available in XML DTD, SOX Schema, and XDR Schema form. The only differences between the xCBL documents in the three formats are the way the validation structures are referenced.

We will again use the extract of parties in a purchase order to compare the semantic capabilities of UN/EDIFACT and xCBL. Furthermore, we have a look at the same examples used to describe the shortcomings of UN/EDIFACT in the previous subchapter in order to see how xCBL is capable of handling the semantics. For our purpose it is not important to look at a certain schema description language supported by xCBL, but at the business content definitions supported by xCBL. In Figure 2 we use the schema-neutral diagram of the xCBL Interconnectivity Guide to show the definition of parties in a purchase order. This diagram is read from top to bottom and from left to right. We have extended the notation to include data type definitions and occurrence indicators, as well as specializations and extensions of an existing element.

The comparison with the examples of the previous subchapter leads to the following results:

- xCBL explicitly mentions the buyer, the seller, the party the goods to ship to, and the party to be billed as meaningful parties in a purchase order. According to the definition the buyer and the seller must be included in the document - there is no other choice even if there is no business need to specify e.g. the seller. The *ShipToParty* and the *BillToParty* are optional. There are no explicit rules for their usage, but for a human the rules seem to be straightforward. In addition to the four party types mentioned, further parties could be used in a document in a list of ’coded’ parties. There already exists an enumeration of different useful types (like StoreNumber or SupplierAgent). But there are no rules concerning the usage of this additional types.
- The assignment of multiple addresses to a party is done - as it could also be done in UN/EDIFACT - by specifying another (‘virtual’) party for each address, especially for the shipping and billing address. Specifying multiple addresses of the same type to a party is only possible for the most common case of multiple delivery addresses. Note, that this is not done in the depicted subtree, but there exists the possibility to specify a final recipient for each order line.

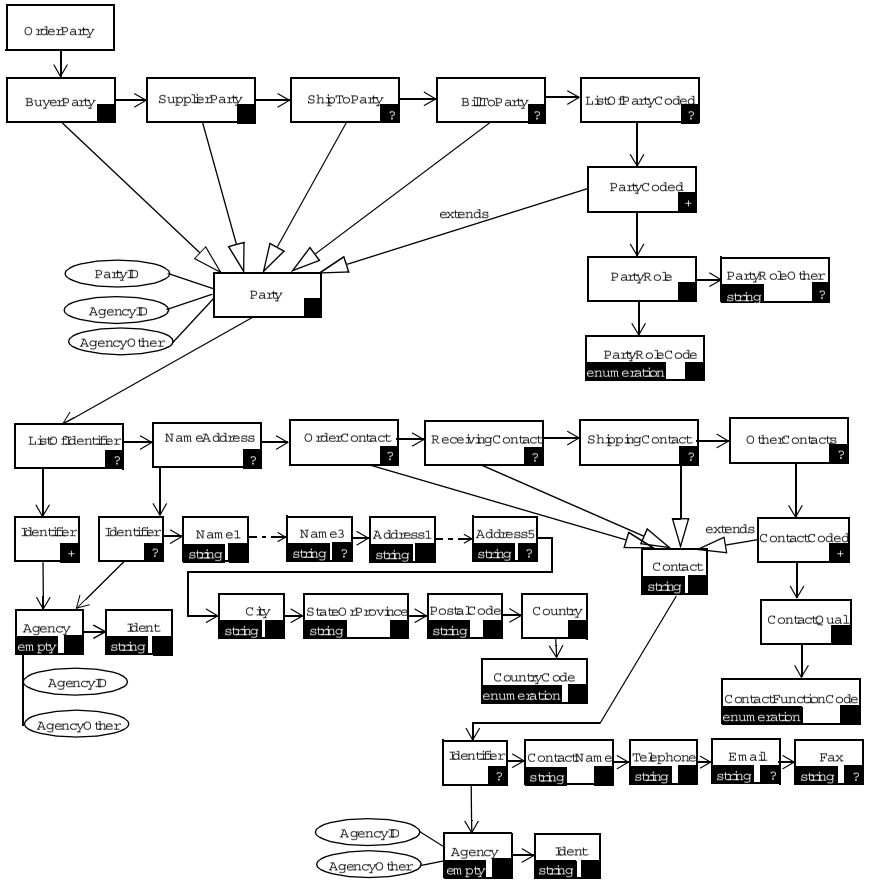

Figure 2 Extract of the Purchase Orders Branching Definition in xCBL 2.0

- In xCBL, there are also multiple ways to identify a party. It is possible to use one or more identifiers for a party. If only one identifier is used it could be specified in the PartyID attribute of Party or in the Identifier element in (a one element) ListOfIdentifiers. If more than one identifier is indicated, the concept of ListOfIdentifiers must be used. The next question is what type of identifiers are used? For this purpose the agency maintaining the identifiers can be given in the AgencyID or AgencyOther Attribute of Party or Agency, respectively. Without prior agreements the receiver must be able to support the identifiers of all the possible agencies.
- Anyway, the usage of identifiers is like in UN/EDIFACT optional. A party could also be specified by its name and address. xCBL defines only a structured way of specifying a name and an address, where all elements are mandatory, except the repeating elements (lines) for the name and the street address.
- Similarly to UN/EDIFACT, contact persons can be included for all types of parties. But a business rule might say that for specific types of business partners a contact has to be specified for the buyer, but is not needed for the supplier. Consequently, also xCBL does not include any constraints to be specified according to data types or specializations.
- The subtree of a party does not include a mechanism for referring documents. Nevertheless, the user of xCBL has also two choices both included in the header information of the purchase order. This is either the reference element (or better the specializations of that element for each party) or the OrderHeaderAttachment in case of referring to an URI.

According to this analysis the definition of an xCBL document is not semantically complete. Although xCBL document definitions are more restrictive and precise than corresponding UN/EDIFACT document definitions they still include a lot of optionality to choose from. Although for xCBL it is true that producing an xCBL-conform document is no guarantee for a receiver supporting xCBL to be able to process it. This means that there is still the need of a more specific definition of the data to be exchanged before business partners can start on-line business transactions.

One might argue that there are other XML-based languages than xCBL which might use a different approach. Some languages with a very specific and limited scope might not have these problems. Nevertheless, any serious language for e-business will have similar problems, because they will take advantage of reusing components. Most of the described problems are a result of reusing different components in a slightly different way.

3. Related work

The idea of e-business interchanges without the need for prior agreements is already documented in the framework of the Open-edi standard. Open-edi work started already in the late 1980ies and became an ISO standard in 1997 (ISO 14662) [15]. One of the key concepts of Open-edi is the distinction between a business operational view (BOV) and a Functional Service View (FSV). The BOV is a perspective of business transactions limited to those aspects regarding the making of business decisions and commitments among organizations. The

FSV is a perspective of business transaction limited to those information technology interoperability aspects of IT systems that are needed to support the execution of Open-edi transactions. The BOV related standards are employed by business users understanding the operating aspects of a business domain or organization, whereas the FSV related standards are used by the information technology experts.

The Techniques and Methodology Working Group (TMWG) of the United Nation's Centre for Trade Facilitation and Electronic Business (UN/CEFACT) is currently working on the OO-edi standard, which follows the Open-edi framework. TMWG thereby defines a methodology called UN/CEFACT's Modeling Methodology (UMM) for defining BOV standards [13][14][22]. This methodology is based on the Unified Modeling Language (UML) [2] and is a customized version of Rational's Unified Process (RUP) [17].

Recognizing the growing popularity of XML and the proposals to use XML in the EDI world [19], UN/CEFACT asked TMWG to research the feasibility of XML to be utilized in B2B electronic commerce. TMWG decided not to follow suite in transforming EDI documents into XML, but instead to use a top-down, business process model driven approach to design business transactions using XML as transfer syntax.

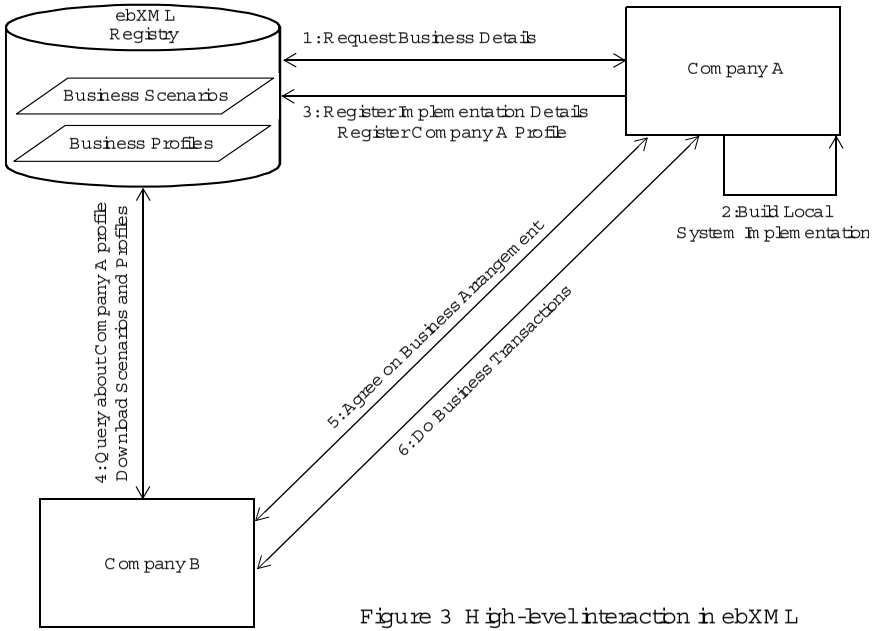

Figure 3 High-level interaction in ebXML

TMWG recommended UN/CEFACT not to do this approach on their own, but to find a partner that would bring in the XML community. After some research UN/CEFACT found OASIS as partner for this approach. UN/CEFACT and OASIS initiated *electronic business XML* (ebXML) a 18 month world-wide project, which started in November 1999. ebXML should lead to standardization of XML business specifications. The purpose of ebXML - as defined in the terms of reference [7] - is to research and identify the technical basis upon which the global implementation of XML can be standardized. The goal is to provide an open technical framework to enable XML to be utilized in a consistent and uniform manner for the exchange of electronic business data in application-to-application, application-to-person and person-to-application environments.

The ebXML Technical Architecture document defines a high-level use case scenario for two trading partners as depicted in Figure 3 [9]: *Company A requests business details from the ebXML registry (step 1) and decides to build its own ebXML compliant application. The Company A submits its own business profile information to the ebXML registry. The business profile submitted to the ebXML registry describes the company's ebXML capabilities and constraints, as well as its supported business scenarios.Company B, which uses an ebXML compliant shrink-wrapped application, discovers the business scenarios supported by Company A in the registry (step 4). Company B sends a request to Company A stating that they would like to engage in an business scenario (step 5). Before engaging in the scenario company B submits a proposed business arrangement directly to Company A's ebXML compliant software interface. The proposed business arrangement outlines the mutually agreed upon business scenarios and specific agreements. Company A then accepts the business agreement.Company A and B are now ready to engage in eBusiness using ebXML (step 6).*

In the following paragraphs we want to roughly summarize those key concepts of ebXML that are important for the approach presented in this paper.

To ensure interoperability, the ebXML initiative has to provide the most common business objects that will be used to assemble documents in a particular business process. Therefore ebXML defines so-called core components. *A core component captures information about a real world business concept, and the relationships between that concept, other Business Information Objects, and a contextual description that describes how a Core or Aggregate Information Entity may be used in a particular ebXML eBusiness scenario* [9].

Accordingly, core components are recursively grouped into information entities, which make up a business document. The extract of the draft version of the ebXML Collaboration Modeling Metamodel in Figure 4 denotes this fact. When business partners collaborate in an ebXML scenario they will perform one

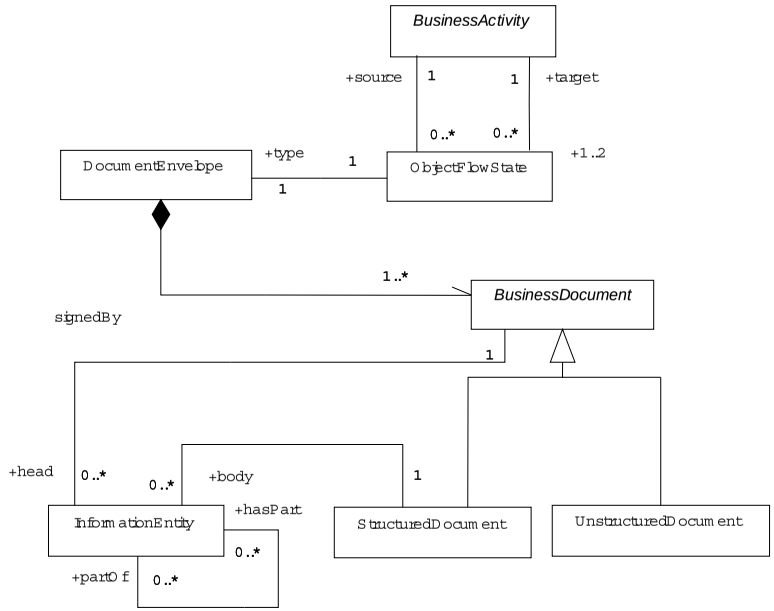

Figure 4 Business Activities and Business Documents in ebXML

or more commercial transactions according to a defined choreography. Each commercial transaction includes a requesting business activity and an optional responding business activity. Business activities are triggered by and result in object flow states. An object flow state is a business event signifying an incoming or outgoing business document.

The defined business collaboration specifications will be stored in a repository. Furthermore, companies publish information about the *roles* they support in specific business collaborations and specific technology implementation details in the collaboration protocol profile (CPP). Consequently, a CPP will explicitly include a link to a business process collaboration. The document exchanges a company has to support are implicitly defined by the documents the role has to support in the collaboration. Companies willing to conduct business have to form a collaboration protocol agreement (CPA) that is derived from the intersection of their *CPP*s.

4. Defining views on business documents

The approach presented in this section is strongly based on the ebXML specification. We assume the existence of reusable common business objects (CBOs). With CBOs we define the object-oriented presentation of core components to be used in information models. A library of CBOs is expected to be delivered by UN/CEFACT continuing the core components effort of ebXML. Since the CBOs do not yet exist, we have 'invented' some for our demonstration purpose. Furthermore, it should be noted that CBOs are on the BOV level of the Open-edi reference model. CBOs are independent of a specific transfer-syntax (e.g. xCBL), but have to be presented in an transfer-syntax to implement the e-business transaction.

The proposed extension to the ebXML specification is based on the following facts: A company defines in its CPP the roles it can fulfil in certain business collaborations. For this purpose the CPP references a business collaboration, which is defined in a process-specification document that conforms to the ebXML business process specification schema specification. Since the reference is on the level of the business collaboration, it follows that a full support of the documents assigned to the business activities in these business transactions is absolutely required. Therefore, there is a high demand in the quality of the specification of the business document types. Business document types must not include a lot of optionality without specifying the conditions for usage. Therefore, the business document type definition must be more specific than those presented in Section 2. Furthermore, there is no chance for slight variations in using the documents among companies playing the same role in a business collaboration.

Business documents are composed of recursively grouped CBOs. Nevertheless, CBOs are usually used differently in different documents or even differently in different occurrences within the same document. Different usage according to a defined set of parameters is focused by the ebXML concept of Context [8]. Note that the presented approach does not stay in contradiction with the Context concept. It could also be used to specify Context rules.

We feel that the current collaboration modeling metamodel does not sufficiently support the definition of unambiguous business documents. It only defines a recursive grouping of CBOs without any additional rules on this grouping. In the following we analyze what additional information is needed to define unambiguous access to business documents. We use a simple example to show how these additional rules can be defined using UML diagrams. Since we feel that the information should go into a registry it would be desirable to define an equivalent (XML-based) representation language capturing all semantics covered by the presented UML diagrams. But we have not yet investigated in such a structure.

To illustrate the importance of unambiguous business document definitions consider the following example: A buyer places a purchase order at a seller and expects a response back. The definition of this business transaction is depicted in Figure 5. This diagram is in-line with the meta model extract of Figure 4. A company willing to purchase goods or services from a buyer would state in its CPP that it is able to perform the buyer role in this transaction (which could be part of an more complex business collaboration). The big question is: What is the information structure of the purchase order and the purchase order response?

We assume the following simplified business requirements for the transaction: The purchase order must have a unique purchase order identification and must include the date the purchase order is placed. The buyer must identify itself either by the buyer identification assigned by the seller or by its full address. This address is also used for billing. The buyer can indicate multiple delivery addresses, P.O. boxes are not useful in a delivery address. Furthermore, the buyer might specify multiple contact persons to be consulted for questions regarding the purchase order. There is no need to state a seller information in the purchase order, but the buyer might send the seller identification assigned by the buyer itself.

If we assume that there exist information entities for date, party, address, contact, and so on, a purchase order document can be assembled by these information entities like in Figure 6. The resulting document structure would be capable of capturing all the business requirements. But this document structure - which reminds one of the structures described in Section 2 - would also cover semantics not part of the requirements specification.

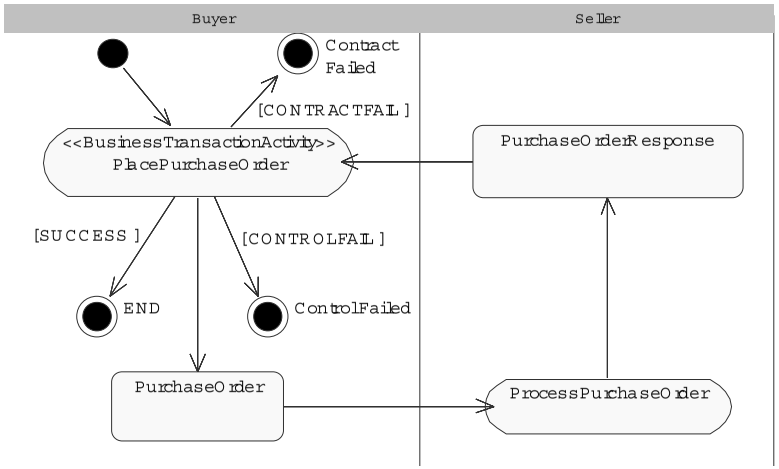

Figure 5 Activity Diagram for the PurchaseOrder Example

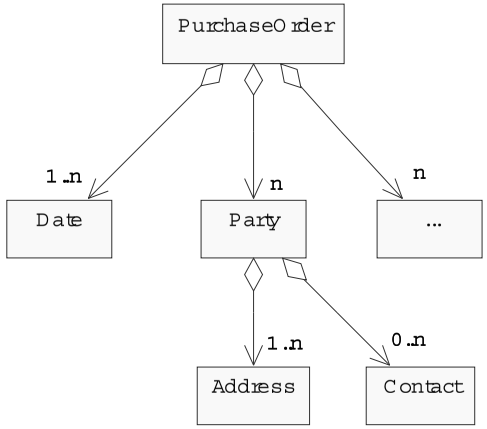

Figure 6 An ambigous PurchaseOrderDocument

One of the major problems of the solution presented in Figure 6 is the fact that some CBOs represent business concepts which stay in conflict with each other. For example, the party CBO is used to present different classes of parties. These different party classes use different composition hierarchies. It would not be useful to define a different CBO for each different class of usage of a party (and all the other CBOs). This would result in a proliferation of similar CBOs. But it would be possible to classify the composition hierarchies according to the roles of the business objects in the aggregation.

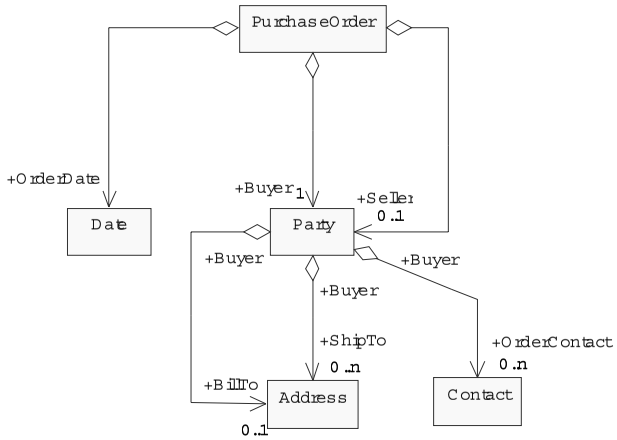

Figure 7 Definition of a PurchaseOrderDocument using Roles

A solution using roles on aggregation hierarchies is presented in Figure 7. According to this definition the purchase order must include an order date and buyer information. The seller information is optional. Addresses and contact information are only relevant for the buyer not for the seller. There could be sent a billing and multiple shipping addresses. All other addresses are not allowed. Similarly, multiple contacts could be stated for the buyer, but only for questions regarding the order (OrderContact).

The use of role-based aggregation hierarchies requires a well defined set of roles a specific CBO can play. It would be even better to define the roles in dependency of relationships to other CBOs. Nevertheless, role-based aggregations are not enough. Not all of the attributes of a CBO will be used in a B2B interchange. Even worse, different roles of the same CBO might access different class attributes. We use collaboration diagrams to show the access to different attributes of CBOs.

So far we have not shown any attributes of the classes in the diagrams of our example. It is not our goal to define the exact CBOs to be used in B2B. Candidates would be the ones developed in the ebXML environment. To further illustrate our example we have made assumptions of reasonable attributes for our CBOs. The attributes given in the simplified class diagram in Figure 8 should not be considered as a complete list.

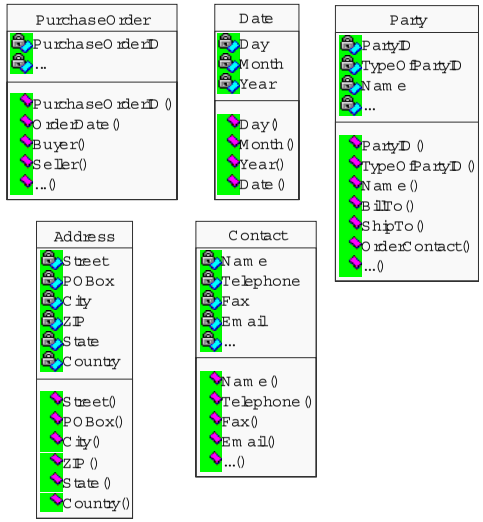

Figure 8 "Simplified" Classes Definitions of CBOs

In a collaboration diagram we show the operation calls to access the attributes. We do not care whether this is a write access of the originator of the document or a read access of the receiver of the document. An operation accessing an attribute is called exactly the same way the attribute is called. Furthermore, it is necessary to show the access to related classes as defined in Figure 7. Therefore, each class includes an operation for each navigable aggregation. Each operation is called by the role of the target class. For example, *Purchase Order* defines navigable aggregation to the target classes *Date* (role: *OrderDate*) and Party (roles: *Buyer* and *Seller*). Consequently, the operations *OrderDate()*, *Buyer()* and *Seller()* are defined in the Class *PurchaseOrder*.

Figure 9 depicts the resulting collaboration diagram for the given business requirements statement. Looking at this collaboration diagram it becomes clear that buyer and seller information are significantly different although both are based on the same CBO *Party*. In our collaboration diagram we use an italic font for optional or conditional operation calls. This means that the originator of a B2B

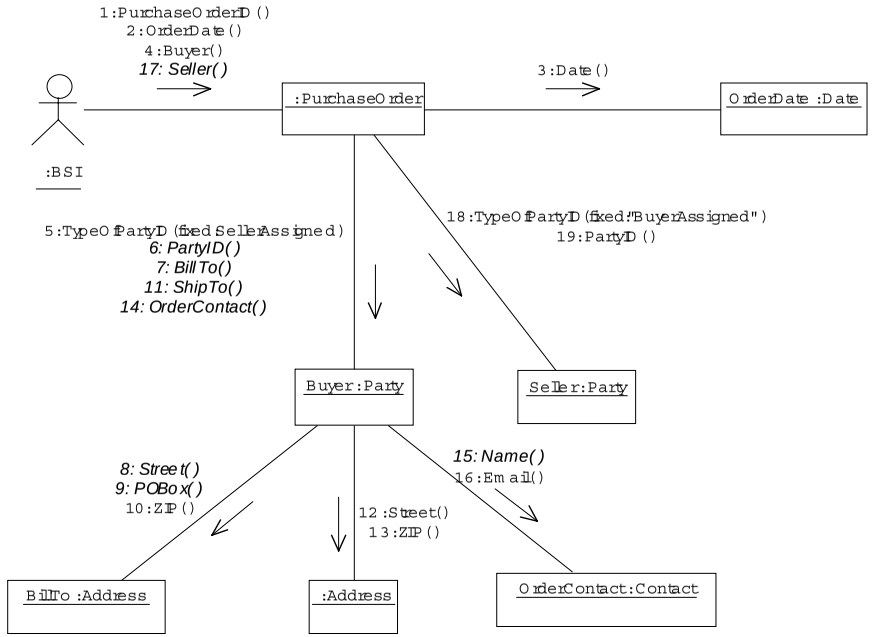

Figure 9 Access to the Attributes of CBOs

document needs not write the corresponding attribute value (or values in case of calls to aggregation classes) and, as a consequence, the receiver cannot always expect a value for this attribute (or these attributes). For example seller information needs not be included in the document. Hence, the Seller() operation of purchase order appears in an italic font. In this case we face real optionality, which means that no rules for applicability exist. In other cases it is not optional, but rather conditions for the usage have to be specified. For example, either the buyer is identified by the seller assigned party identification or the full address (which becomes the billing address) must be transmitted. Both the PartyID() operation and the BillTo() operation are optional or better conditional. However access rules cannot be shown in a collaboration diagram.

In order to define the basic rules for instantiating an activity diagram is used to show if-statements and loops. Basic rules define what happens if a value is defined or not. Furthermore, it can be tested whether an attribute has a certain value or not. As an example consider Figure 10 which defines the inside of the method *Buyer()*. Note that the activity diagrams must not include anything about the application logic of a partner's in-house application. Their only function is to define clear rules for data access in the documents.

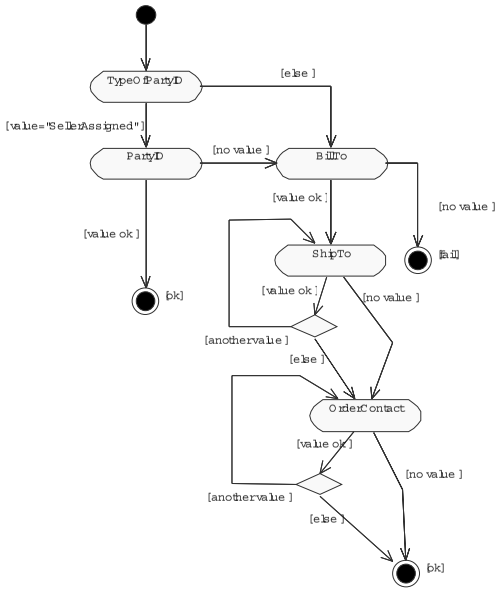

Figure 10 Activity Diagram for Buyer()

The activity diagrams conclude our approach to the definition of unambiguous B2B documents. To summarize, we use role-based aggregation hierarchies, collaboration diagrams and activity diagrams to depict in our approach the B2B documents and their access rules. However, those types of diagrams provide only different viewpoints into the same solution space. Therefore, an (XML) document structure being capable of capturing all the semantics of the solution space (or what is defined in the three diagram types) would be desirable. Of course, UML can always be used as graphical definition and presentation language, but an equivalent XML presentation can be used for registry storage. The enhanced B2B document structure definition should go into definition of business documents supporting business activities in the collaboration modeling metamodel. This would result in an extension to Figure 4.

Speaking in ebXML terms, a more precise business process definition would effect the definition of the possibilities a company has to choose from. But is there also a consequence for the capabilities a company defines in its CPPs or in the agreements of the CPAs? The approach could also be used to redefine the business documents in a company's profile. Of course, the redefinition must not conflict with the original one in the business process definition. For example, a certain company could decide that it will always send the seller information in our example. This would add a company-specific context to the definition of business documents. In this case the CPP needs to reference the business activities and business documents in addition to the more general business processes. CPAs could also be extended to include valid intersections of business document definitions.

5. Conclusion

In this paper we describe an approach to specify different views on documents used in B2B transaction. The approach is motivated by the fact that traditional standardization approaches - UN/EDIFACT, but also many current XML efforts - offer a document structure and a vocabulary that covers a whole lot more than is actually needed for an interchange by an organization. Unfortunately, the standardization approaches stop at this step. Trimming down the document structure to the actual user needs is out of scope of traditional approaches. But a lot of effort - multiple times more than that of standardization - is spent on this activity. Since no rules for this customization exist, this activity is the major reason for incompatibility.

A generic approach to describe document structures leaves too much scope for misinterpretation and for document instances that are not conform to the

original business goals. This commonly results from the reuse of components in the document definitions, but having different access rules to their instances based on certain criteria. These criteria are often based on different roles a component can play or on data values in other instances. The rules on how to access a document are in traditional EDI known as message implementation guidelines (MIGs). There must be an agreement on these message implementation guidelines before business can start exchanging business documents.

In our paper we present a first step towards machine-readable MIGs. In fact, we use a set of UML diagrams to define document structures and rules to access these structures. The approach is strongly based on the concept of the ebXML initiative and on the usage of common business objects. In ebXML a business transaction between two or more organizations consists of multiple activities. An activity might be triggered by an input document and results in an output document. A document is composed of recursively grouped information entities, which are common business objects in a certain context. We claim that without specifying exact access rules to the document structures, it would be possible to create valid document instances which are by definition conform to the document type, but are not conform to the business requirements. The presented approach defines access rules to these information entities which follow the document type definitions as well as the business requirements. For example, it is possible to define - according to business requirements - different access rules to buyer and seller information although both are based on the same CBO concept of party information. The approach uses UML class diagrams utilizing roles on the association of aggregation hierarchies, collaboration diagrams and activity diagrams.

In a future step the semantics captured in these diagram types must be translated into a representation format for storage in a registry. Ideally, this document format itself is based on XML. This would allow an application to query the exact rules to create an outgoing business document and parse an incoming business document. Applications could verify whether they can support the rules or not, which would be a major step into the direction of automating the agreement process.

References

[1] Berge, J., "The EDIFACT Standards", NCC Blackwell Ltd., Oxford, (1991)

[2] Booch, G., Jacobson, I., Rumbaugh, J., "The Unified Modeling Language User Guide", Addison Wesley Longman, Reading, (1998)

[3] Burt, C. (ed.), "OMG BOMSIG survey with published definition of a business object", OMG document 95-02-04, http://www.omg.org

[4] Casanave, C., "Business-Object Architectures and Standards", Proceedings of OOPSLA '95, Workshop on Business Objects Design and Implementation, Springer, London, (1995)

[5] Commerce One, "XML Common Business Library", http://www.xcbl.org

[6] Cover, R., "The Essence and Quintessence of XML" Retrospects and Prospects, OASIS, http://www.oasis-open.org/html/essence_of_xml.html

[7] ebXML, "ebXML - Terms of Reference", ebXML, http://www.ebxml.org, (1999)

[8] ebXML, "ebXML Core Components", http://www.ebxml.org/project_teams/core_components/core_components.htm

[9] Eisenberg, B. and Nickull D., "ebXML Technical Architecture Specification v1.0.4", http://www.ebxml.org, (2001)

[10] Emmelhainz, M.A., "Electronic Data Interchange: A Total Management Guide", Van Nostrand Reinhold, New York, (1990)

[11] Feldman, S., "The Changing Face of E-Commerce: Extending the Boundaries of the Possible", IEEE Internet Computing, May/June (2000)

[12] Hill, N.C. and Ferguson, D.M., "Electronic Data Interchange: A Definition and Perspective", EDI Forum: The Journal of Electronic Data Interchange, vol. 1, no. 1, (1989)

[13] Huemer, C., "Modeling Inter-Organizational Systems with UML", Proceedings of the 12th International Electronic Commerce Conference, Bled (Slovenia), (1999)

[14] Huemer, C., "Defining Electronic Data Interchange Transactions with UML", Proceedings of HICSS-34 (34th Hawaiian International Conference on System Sciences), Maui, 2001

[15] ISO, "The Open-edi Reference Model", ISO/IEC JTC1/SC30 IS 14662, (1996)

[16] Kotok, A., "Extensible and More - A Survey of XML Business Data Exchange Vocabularies", http://www.xml.com/pub/2000/02/23/ebiz/index.html,

[17] Kruchten, P., "Rational Unified Process", Addison-Wesley Object Technology Series, Reading, (1998)

[18] OMG, "An Overview to the XMI - XML Metadata Interchange Specification", OMG, http://www.omg.org/news/pr99/xmi_overview.html

[19] Peat, B. and Webber, D, "Introducing XML/EDI - the E-business framework", http://www.geocities.com/WallStreet/Floor/5815/start.html

[20] Raman, D., "XML/EDI", Cyber Assisted Business in practice, TIE Holding NV, Am Hoofdorp (Netherlands), (1999)

[21] TMWG, "Review of the Current EDI Standardisation Process", CEFACT/TMWG/N021

[22] TMWG, "UN/CEFACT's Modelling Methodology (UMM)", CEFACT/TMWG/N090/R9

Biographical Sketch

Christian Huemer received the PhD degree in information systems from the University of Vienna in 1993. Currently, he is an assistant professor at the Institute for Computer Science and Business Informatics of the University of Vienna, Austria. His main research areas are methodologies for modeling e-business transactions as well as meta data support for electronic business and EDI environments. He is a member of the United Nation's Center for Trade Facilitation and Electronic Business (UN/CEFACT). Under the umbrella of UN/CEFACT he serves as Secretary of UN/CEFACT's Techniques and Methodology Working Group (TMWG) and was responsible for the OO-edi Demo Project of the EDIFACT Working Group. He is a member of the ebXML initiative and hosted the final ebXML Meeting in May 2001.