Registering UMM Business Collaboration Models in an ebXML Registry

Birgit Hofreiter*, Christian Huemer* and Marco Zapletal°
*Faculty of Computer Science, University of Vienna, Liebiggasse 4, 1010 Vienna, Austria
°Faculty of Informatics, Vienna University of Technology, Favoritenstrass 9-11, 1040 Vienna, Austria
{birgit.hofreiter, christian.huemer}@univie.ac.at, marco@ec.tuwien.ac.at

Abstract

UN/CEFACT's modeling methodology (UMM) is used to develop global choreographies of inter-organizational business processes. UMM models should be publically available in order to foster re-use and to reference them in trading partner agreements. In this paper we define a mapping of UMM models or parts thereof to the ebXML registry information model (RIM).

1 Motivation

A succesful B2B implementation requires not only commonly accepted document types, but also an agreed choreography of the supported business proccesses. A choreography describes the flow of interactions between the participating business partners that interlink their individual processes. We distinguish local and global choreographies. A local choreography describes the flow from a participating partner's point of view. It makes the public parts of its local process visible to others. A global choreography defines the inter-organizational process from a neutral perspective.

In practice, we see two different alternatives to realize inter-organizational business processes. Firstly, one business partner announces his public choreography. All other business partners that want to collaborate must exactly meet the complementary role(s) and adapt their interfaces accordingly. Discovering potential business partners requires complex comparisons of public choreographies. The second approach is based on well accepted global choreographies. Standards organizations, industry consortia or market leaders define a global choreography. An enterprise supporting a standard global choreography has to derive the public choreography of the supported role and bind its interfaces accordingly. The complexity of discovering potential business partners is reduced to finding partners supporting a complementary role in a global choreography.

In this paper we follow the idea of the second alternative. In particular we concentrate on the registration of global choreographies. The registration of global choreographies supports three aspects in inter-organizational business processes. Firstly, it helps enterprises to search for global choreographies that are worth to support in future. In order to facilitate the search it is important to classify the choreography according to its business environment (e.g. geopolitical, industry sector, etc.). Secondly —once the enterprise has implemented the choreography—the enterprise searches for business partners supporting a complementary role of the same choreography in exactly the same business context. Finally, the registration of choreographies is also useful for the standard setting organizations. New choreographies under development may re-use other choreographies or parts thereof.

2 UMM

For the purpose of modeling global choreographies we use UN/CEFACT's modeling methodology (UMM). UMM is specially designed to describe inter-organizational business processes from a global perspective. It defines a UML profile—i.e. a set of stereotypes, tagged values and constraints—in order to customize the UML meta model for the special purpose of modeling the collaborative space in B2B [18].

The UMM uses three main views in order to create so-called business collaboration models. The ***business domain view*** is used to gather existing knowledge. Business processes are discovered not constructed. This helps identifying possible collaborations in the next step, the ***business requirements view***. Use cases and associated worksheets are used to collect the requirements of identified business collaborations. The ***business transaction view*** covers the analysis model defining the choreography and the information exchanged. It delivers of three main types of artefacts: business collaboration protocols, business transactions and business documents. A business transaction is a unit of work that leads to a sychronized business state in the information systems of two collaborating partners. It is realized by exchanging a business document and optionally another one representing the response. A business collaboration protocol is a choreographed set of business transactions.

Due to space limitations we are not able to present all the UMM artefacts in detail. The interested reader is refered to the following publications: [4,18]. We will demonstrate the registration of UMM models by means of a *request for quote* business transaction (see Figure 1). A UMM transaction follows always the same pattern due to

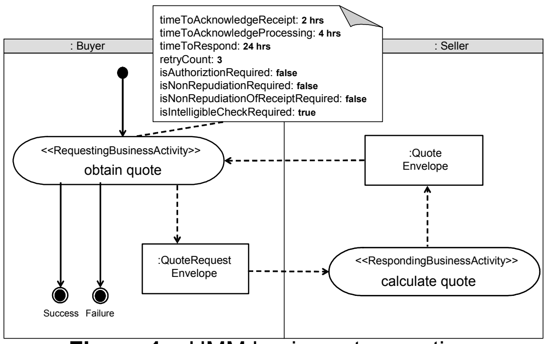
timeToAcknowledgeReceipt: 2 hrs
timeToAcknowledgeProcessing: 4 hrs
timeToRespond: 24 hrs
retryCount: 3
isAuthoriztionRequired: false
isNonRepudiationRequired: false
isNonRepudiationOfReceiptRequired: false
isIntelligibleCheckRequired: true
: Buyer
: Seller
<<RequestingBusinessActivity>>
obtain quote
:Quote Envelope
:QuoteRequest Envelope
<<RespondingBusinessActivity>>
calculate quote
Success
Failure

Figure 1. UMM business transaction

its strict definition: A business transaction is always performed between two business partners that are assigned to exactly one swimlane each. Each partner performs exactly one activity. An object flow between the requesting and the responding business activity is mandatory. An object flow in the reverse direction is optional. In the *request for quote* business transaction of Figure 1 the *buyer* performs *obtain quote*, which outputs a *quote request envelope*. This envelope is input to the *calculate quote* activity executed by the *seller*. Since the finale state depends on a decision of the *seller*, a *quote envelope* is returned. Note that UMM elements usually carry tagged values. In Figure 1 we show only those of *obtain quote* which are also used in the following sections.

3 Mapping UMM Models to ebXML RIM

3.1. Representing UMM Models in XMI

In order to register UMM models the graphical notation must be expressed in a machine-readable format. Since UMM is based on UML it seems to be straight-forward to use OMG's XML Metadata Interchange (XMI) [12]. The main goal of XMI is—according to its documentation—enabling easy interchange of meta-data between modeling tools and between tools and meta-data repositories in distributed heterogeneous environments. XMI is based on XML, UML and the Meta Object Facility (MOF) [13].

For the purpose of representing UMM models in XMI we have to focus on two different MOF levels: the UMM profile itself on the meta-model layer (M2) and the business collaboration model on the model layer (M1). The UMM profile—to which we contributed major parts—defines a set of UML stereotypes and their tagged values as well as UMM-specific constraints on the UML meta model. An XMI representation of the UMM profile allows an import of the UMM profile into UML modeling tools. This means the UML tool imports the stereotypes and the model is able to use them in creating a business collaboration model. Thus, we have created an XMI version of the UMM profile which we also use in our UMM extension to the UML tool *Enterprise Architect*.

The XMI on Level M2 defines all stereotypes by a unique id and a name. In order to demonstrate the XMI definition of stereotypes we have selected three stereotypes in the code below. This code of the M2 XMI file is based on XMI version 1.2 for UML version 1.4. Line 1 starts the definition of the stereotype *business action* and line 12 the one of stereotype *requesting business activity*. The definition of stereotype *business transaction* is roughly pointed out in line 19. All other UMM stereotypes are not shown here. In order to improve readability we have replaced the values of all unique id's by human readable identifiers throughout the paper. The first sub-element of *stereotype* is used to define the *base class* of the stereotypes. In case of *business action* and *requesting business activity* this is the UML meta class *action state* (lines 2 and 13). If fact, *business action* is the superclass of *requesting business activity* (and *responding business activity* which is not outlined in the code). This generalization is defined in line 23.

Thus, *requesting business activity* inherits all tag definitions of *business action*. Tag definitions specify new kinds of properties extending a stereotyped model element. It can be compared to meta-attribute definitions [3]. The list of tag definitions characterizing a business action is listed in lines 3 to 10. These characteristics are *time to acknowledge receipt*, *time to acknowledge processing*, *is non-repudiation required*, *is non-repudiation of receipt required*, *is intelligible check required* and *is authorization required*. Most of these tag definitions are self-explanatory. An *acknowledgment of receipt* is sent after grammar validation, sequence validation, and schema validation. However, if the *is intelligible check required* flag is false, the acknowledgment is sent after receipt without any validation. An *acknowledgment of processing* is sent after the business information passes aset of business rules and is handed over to the application for processing.

```
[1]  <Stereotype xmi.id="st_BusinessActionType" name="BusinessAction">
[2]     <Stereotype.baseClass>ActionState</Stereotype.baseClass>
[3]     <Stereotype.definedTag>
[4]        <TagDefinition xmi.id="tv_timeToAcknowledgeReceiptType"
              name="timeToAcknowledgeReceipt" tagType="Time"/>
[5]        <TagDefinition xmi.id="tv_timeToAcknowledgeProcessingType"
              name="timeToAcknowledgeProcessing" tagType="Time"/>
[6]        <TagDefinition xmi.id="tv_isAuthorizationRequiredType"
              name="isAuthorizationRequired" tagType="Boolean"/>
[7]        <TagDefinition xmi.id="tv_isNonRepudiationRequiredType"
              name="isNonRepudiationRequired" tagType="Boolean"/>
[8]        <TagDefinition xmi.id="tv_isNonRepudiationOfReceiptRequiredType"
              name="isNonRepudiationOfReceiptRequired"
              tagType="Boolean"/>
[9]        <TagDefinition xmi.id="tv_isIntelligibleCheckRequiredType"
              name="isIntelligibleCheckRequired" tagType="Boolean"/>
[10]    </Stereotype.definedTag>
[11] </Stereotype>
[12] <Stereotype xmi.id="st_RequestingBusinessActivityType"
        name="RequestingBusinessActivity">
[13]    <Stereotype.baseClass>ActionState</Stereotype.baseClass>
[14]    <Stereotype.definedTag>
[15]       <TagDefinition xmi.id="tv_retryCountType"
              name="retryCount" tagType="Integer"/>
[16]       <TagDefinition xmi.id="tv_timeToRespondType"
              name="timeToRespond" tagType="TimeExpression"/>
[17]    </Stereotype.definedTag>
[18] </Stereotype>
[19] <Stereotype xmi.id="st_BusinessTransactionType"
        name="BusinessTransaction">
[20]    <Stereotype.baseClass>Activity Graph</Stereotype.baseClass>
[21]    ...
[22] </Stereotype>
[23] <Generalization xmi.id="..." child="st_RequestingBusinessActivityType"
        parent="st_BusinessActionType"/>
```

In addition to these inherited tag definitions, *requesting business activity* has its own specific tag definitions *retry count* and *time to respond* listed in lines 14 to 17. *Retry count* refers to the number of tries to re-initiate a business transaction in case of control failures. *Time to respond* is the maximum time to receive a responding business document.

The XMI M2 model of the UMM profile serves as a kind of reference for the XMI files containing the business collaboration models. This means model elements of the UMM XMI file at M1 will refer to the id of the corresponding stereotype of the M2 model. The code in lines 24 to 84 represents the business transaction *request for quote* depicted in Figure 1. In line 24 starts the definition of the activity graph with the id *BT1* named *request for quote*. Since the activity graph is stereotyped as *business transaction*, the id in the *stereotype* attribute in line 24 references the corresponding *stereotype* element in line 19 of the M2 XMI file.

In UML version 1.4 activity graphs are specializations of state machines. The whole activity graph is embedded into a single virtual overall composite state. Line 25 navigates to this top state which definition starts in line 26. This composite state includes all states of our business transaction shown in Figure 1: It covers two action states—*obtain quote* (*BA1* in line 28) and *calculate quote* (*BA2* in line 46). Furthermore, it is composed of the object flow states representing the *quote request* envelope (*Info1* in line 44) and the *quote* envelope (*Info2* in line 45). Finally, it includes the initial state and the two final states (lines 51 - 53).

We learned that a requesting business activity is characterized by 8 tag definitions. Accordingly, the *obtain quote* activity instantiates a tagged value for each tag definition. The list of tagged values is specified in lines 29 to 42. Each tagged value has its own id and its *type* attribute references the id of the corresponding tag definition in the M2 XMI file (lines 4 to 9 or lines 15 to 16). The tagged values of the *calculate quote* activity have been truncated due to space limitations.

Furthermore, our activity graph is composed of partitions *BTS1* (representing the role *buyer*) and *BTS2* (representing the role *seller*) and its corresponding transitions *T1 - T7*. These transitions show the flow within the business transaction. Four out of seven build the object flow within the transaction whereas the remaining three transitions lead from the initial state or to the final state.

Finally, the partition *BTS1* (lines 67 - 80) consists of the action state *BA1*, the object flow state *Info1*, the initial state *PS1*, the final states *FS1* and *FS2* as well as of the transitions *T1*, *T2*, *T3*, *T6*, and *T7*. Note, that *T3* and *T5* are transitions from one partition to another. They are assigned to the partition from which they start. Therefore *T3* belongs to *BTS1* and *T5* to *BTS2*. The partition of the role seller *BTS2* has been truncated of our XMI code again due to space limitations. It would cover the action state *BA2*, the object flow state *Info2*, and the two remaining transitions *T4* and *T5*.

```
[24] <ActivityGraph xmi.id="BT1" name="request for quote"
         stereotype="st_BusinessTransactionType">
[25]   <StateMachine.top>
[26]     <CompositeState xmi.id="...">
[27]       <CompositeState.subvertex>
[28]         <ActionState xmi.id="BA1" name="obtain quote"
                 stereotype="st_RequestingBusinessActivityType">
[29]           <ModelElement.taggedValue>
[30]             <TaggedValue xmi.id="TV01"
                   type="tv_timeToAcknowledgeReceiptType">
[31]               <TaggedValue.dataValue>PT2H</TaggedValue.dataValue>
                 </TaggedValue>
[32]             <TaggedValue xmi.id="TV02"
                   type="tv_timeToAcknowledgeProcessingType">
[33]               <TaggedValue.dataValue>PT4H</TaggedValue.dataValue>
[34]             </TaggedValue>
[35]             <TaggedValue xmi.id="TV03"
                   type="tv_timeToRespondType">
[36]               <TaggedValue.dataValue>PT24H</TaggedValue.dataValue>
[37]             </TaggedValue>
[38]             <TaggedValue xmi.id="TV04"
                   type="tv_retryCountType">
[39]               <TaggedValue.dataValue>3</TaggedValue.dataValue>
[40]             </TaggedValue>
[41]             <!-- Tagged Values for T05 - T08
                      isAuthorizationRequired (false),
                      isNonRepudiationRequired (false),
                      NonRepudiationOfReceiptRequired (false)
                      isIntelligibleCheckRequired (true) truncated -->
[42]           </ModelElement.taggedValue>
[43]         </ActionState>
[44]         <ObjectFlowState xmi.id="Info1"
                 stereotype="st_RequestingInformationEnvelopeType"
                 type="QuoteRequestEnvelope"/>
[45]         <ObjectFlowState xmi.id="Info2"
                 stereotype="st_RespondingInformationEnvelopeType"
                 type="QuoteEnvelope"/>
[46]         <ActionState xmi.id="BA2" name="calculate quote"
                 stereotype="st_RespondingBusinessActivityType">
[47]           <ModelElement.taggedValue>
[48]             <!-- Tagge Values truncated -->
[49]           </ModelElement.taggedValue>
[50]         </ActionState>
[51]         <Pseudostate xmi.id="PS1" kind="Initial"/>
[52]         <FinalState xmi.id="FS1" name="Success"/>
[53]         <FinalState xmi.id="FS2" name="Failure"/>
[54]       </CompositeState.subvertex>
[55]     </CompositeState>
[56]   </StateMachine.top>
[57]   <StateMachine.transitions>
[58]    <Transition xmi.id="T1" source="PS1" target="BA1"/>
[59]     <Transition xmi.id="T2" source="BA1" target="Info1"/>
[60]     <Transition xmi.id="T3" source="Info1" target="BA2"/>
[61]     <Transition xmi.id="T4" source="BA2" target="Info2"/>
[62]     <Transition xmi.id="T5" source="Info2" target="BA1"/>
[63]     <Transition xmi.id="T6" source="BA1" target="FS1"/>
[64]     <Transition xmi.id="T7" source="BA1" target="FS2"/>
[65]   </StateMachine.transitions>
[66]   <ActivityGraph.partition>
[67]     <Partition xmi.id="BTS1">
[68]       <Partition.classifierRole>
[69]         <ClassifierRole xmi.id="...">
[70]           <ClassifierRole.base>
[71]             <ModelElement xmi.idref="IdOfAuthorizedRoleBuyer"/>
[72]           </ClassifierRole.base>
[73]         </ClassifierRole>
[74]       </Partition.classifierRole>
[75]       <Partition.contents>
[76]         <ModelElement xmi.idref="PS1"/>
[77]         <ModelElement xmi.idref="BA1"/>
[78]         <!-- T1, Info1, T2, T3, FS1, FS2, T6, T7 truncated -->
[79]       </Partition.contents>
[80]     </Partition>
[81]     <Partition xmi.id="BTS2">
[82]       <-- Partition of Role Seller including Info2, T4, T5 truncated-->
[83]     </Partition>
[84]   </ActivityGraph.partition>
[85] </ActivityGraph>
```

We used the *request for quote* business transaction in order to demonstrate the XMI presentation of UMM artefacts, since it is used for demonstration throughout the paper. Of course other UMM artefacts not detailed in this paper may be mapped to XMI as well.

3.2. Classifying UMM Models in a Registry

In the previous sub-section we demonstrated the presentation of UMM models or parts thereof in a machine readable format. In this sub-section we concentrate on how to classify these models within the ebXML registry. This facilitates the search for models and enables re-use.

It is envisioned that key players in a business domain will develop UMM business collaboration models. These key players might be standardization bodies like UN/CEFACT itself, industry groups like EAN/UCC, SWIFT, ad-hoc groups for specific processes, or even market leaders in certain domains. In order to make these business collaboration models accessible to the public, the business collaboration models must be stored in registries. As a consequence, potential business partners are able to search for adequate business collaboration models within such registries. Business partners are only interested in models that are valid in the business environment they are operating in. Thus, a business collaboration model must be classified in the registry according to its business context. This means that the body who developed the business collaboration model must bind it to its business environment(s).

The best way to describe a business environment is by the concept of business context as introduced by ebXML core components [17]. In this specification business context is defined as a mechanism for qualifying and refining core components according to their use under particular business circumstances. We enlarge the scope of this definition to apply the mechanism not only to core components but to any UMM artefact. The business context in which the business collaboration takes place is specified by a set of categories and their associated values. In ebXML eight categories have been identified: business process, product classification, industry classification, geopolitical, official constraints, business process role, supporting role, and system capabilities. The context categories are not limited to the ones identified, but we do not recommend the use of other categories. The use of these context drivers or a combination out of these context categories provides an opportunity to bind business process models to a business environment in UMM.

Assume that our *request for quote* transaction was developed by the US print media for ordering books. For the sake of re-use, a model should be valid in multiple business environments. Assume that the Austrian computer industry finds this *request for quote* transaction in a registry and recognizes that this model exactly fits its needs. Note, that the examples in this paper use only a combination of two context drivers—product classification and geopolitical—to avoid unnecessary complexity in demonstration.

The context binding may be realized by two concepts: The first option is a classification within the model itself [4,6]. This means the stereotypes of a UMM model are extended by tag definitions describing the business environment. This approach has a major disadvantage. Whenever a classification is added, deleted or updated it requires a change of the tagged values. It follows that the model is changed although the choreography does not change. This creates a useless number of versions of the same UMM model. The alternative is a context binding external to the model. Metaphorically, each organization that accepts the model for their environment puts a sticker on the model. The US print media creates the model and puts a sticker on it. Later on, the Austrian computer industry finds the model and puts another sticker on it. If the US print media updates the model it creates a new version of the model and moves the sticker from the old to the new version. It is up to the Austrian computer industry to leave the sticker on the old version or to move it to the new one.

In the electronic world this concept must be realized by the registry meta-data assigned to a business collaboration model. In the ebXML registry information model (RIM) a UMM model is stored as an *extrinsic object*. Multiple classifications can be assigned to such an *extrinsic object*. The code in lines 86 to 103 shows the classification of our *request for quote* transaction to the two business environments outlined above. The *extrinsic object* defines a unique id for the UMM model, which may correspond to the location of the UMM XMI file. Unfortunately, ebXML RIM does not provide the concept of *category groups* known from UDDI [11]. We need such a concept to create a group *US* and *printed publication* and another one *Austria* and *computer* avoiding invalid Cartesian product combinations like *Austria* and *printed publication*. As a work-around we use the ebXML RIM concept of *slots*. An *extrinsic object* includes a slot for each business environment. A *slot* is built by a *value list*. The value list includes a *value* for each context driver. Each value references a classification.

```
[86]  <ExtrinsicObject
         id="http://www.whoever.org/requestForQuote.xmi"
         mimeType="text/xml" objectType="urn:uncefact:umm">
[87]        <Slot name="urn:someone:classficationGroup1">
[88]           <ValueList>
[89]              <Value>urn:someone:umm:classification1</Value>
[90]              <Value>urn:someone:umm:classification2</Value>
[91]           </ValueList>
[92]        </Slot>
[93]        <Slot name="urn:somebody:classficationGroup2">
[94]           <ValueList>
[95]              <Value>urn:somebody:umm:classification3</Value>
[96]              <Value>urn:somebody:umm:classification4</Value>
[97]        </ValueList>
[98]        </Slot>
[99]        <Classification id="urn:someone:umm:classification1"
               classificationScheme="urn:undp:categorization:unspsc"
               classificationNode="55.10.15.00"/> <!-- Printed Publications -->
[100]       <Classification id="urn:someone:umm:classification2"
               classificationScheme="urn:iso:categorization:iso3166"
               classificationNode="US"/>
[101]       <Classification id="urn:somebody:umm:classification3"
               classificationScheme="urn:undp:categorization:unspsc"
               classificationNode="43.17.18.00"/> <!-- Computers -->
[102]       <Classification id="urn:somebody:umm:classification4"
               classificationScheme="urn:iso:categorization:iso3166"
               classificationNode="AT"/>
[103]</ExtrinsicObject>
```

4 Related Work

In addition to UMM other approaches use UML to model inter-organizational business processes. In the standards world the development of RosettaNet Partner Interface Processes (PIPs) [14] is based on global choreographies described by UML. In the academic world UML is mostly used to conceptualize the public choreography of Web Services, but not the global one [9,15].

In the Web Services world a lot of different languages have been developed to capture business processes, e.g. Business Process Modeling Language (BPML)[1] and Business Process Execution Language (BPEL) [2]. These languages are limited to describe orechestrations and local choreographies. Nevertheless, local choreographies described in these languages may be derived from our global UMM choreographies. In [5] we describe such a mapping from UMM to BPEL.

More recently a draft of the Web Services Choreography Description Language (WS-CDL) [8] has been released in order to complete the family of Web Services standards by a specification supporting global choreographies. Within the ebXML framework, the business process specification schema (BPSS) [16] always describes the choreography of a business collaboration from a global perspective. In our approach the extrinsic object describing the business process is an UMM XMI file. The approach may be easily adapted for extrinsic objects carrying WS-CDL or BPSS data as well—but only if no business environment specific variations exist.

In this paper we used the ebXML Registry and Repository [10]. The Web Services registry standard is UDDI [11]. Although their functionality is quite similar the information models of both standards are significantly different. In [7] we show how to register business collaboration models in UDDI.

5 Conclusion

In this paper we presented an approach to register UMM business collaboration models or parts thereof in an ebXML registry. Registration requires a machine-readable format. Since UMM is based on UML, we have to map the UML graphical notation into a commonly accepted machine-readable format. Consequently, we defined an XMI representation for UMM models. We demonstrated the XMI approach for both the meta-model (M2) and the model (M1) layer of MOF: The meta-model layer defines all stereotypes and associated tagged values of UMM's UML profile. The business collaboration models on the M1 layer reference the corresponding stereotypes.

In a next step we demonstrated the registration of the business collaboration models. These models are managed as extrinsic objects in the ebXML registry. We explained how to classify a model according to its business context. Since a model may be valid in multiple business environments, we developed an approach using the registry meta-data for assigning multiple business environments to the same model. Since the ebXML registry does not natively include any classification groups we presented an approach that is compliant to ebXML RIM.

References

[1] Arkin, A.: Business Process Modeling Language, Version 1.0., BPMI (2002); http://www.bpmi.org/bpml-spec.esp

[2] Andrews, T., et al.: Business Process Execution Language for Web Services, V. 1.1. (2003), http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnbizspec/html/bpel1-1.asp

[3] Booch, G., Jacobson, I., Rumbaugh, J.: The Unified Modeling Language User Guide. Addison Wesley Object Technologies Series, Reading, 1998

[4] Hofreiter, B., Huemer, C., Winiwarter, W.: OCL-Constraints for UMM Business Collaborations, Proceedings the 5th Int'l Conf. on Electronic Commerce and Web Technologies (EC-Web 2004); Springer LNCS, Zaragoza (Spain), Aug. 2004

[5] Hofreiter, B., Huemer, C.: Transforming UMM Business Collaboration Models to BPEL. Proc. of OTM Workshops 2004. Springer LNCS, Vol. 3292, (2004) 507-519

[6] Hofreiter, B., Huemer, C.: From multi-context business collaboration models to context-specific ebXML BPSS. Proc. of IEEE Int'l Conf. on e-Technology, e-Commerce and e-Services (EEE) 2005; IEEE CS; Hong Kong, March 2005

[7] Hofreiter, B., Huemer C.: Registering a Business Collaboration Model in Multiple Business Environments. Proc. of OTM Workshops 2005. Springer LNCS, Vol. 3762

[8] Kavantzas, N. et al: Web Services Choreography Description Language, Version 1.0. W3C (2004) http://www.w3.org/TR/ws-cdl-10

[9] Kramler, G., Kapsammer, E., Retzschitzegger, W., Kappel, G.: Towards Using UML 2 for Modelling Web Services Collaboration Protocols. 1st Int. Conf. on Interoperability of Enterprise Software and Applications.

[10]OASIS: ebXML Registry Information Model v2.0. (2001) http://www.ebxml.org/specs/ebrim2.pdf

[11]OASIS: UDDI Version 3.0.2. http://uddi.org/pubs/uddi_v3.htm

[12]OMG: "XML Metadata Interchange (XMI) Specification, Version 1.4", http://www.omg.org/cgi-bin/doc?formal/02-01-01

[13]OMG, "Meta-Object Facility (MOF), Version 1.4", http://www.omg.org/technology/documents/formal/mof.htm

[14]RosettaNet: RosettaNet Implementation Framework, Core Specification V02.00.01. (2002) http://www.rosettanet.org

[15]Thöne, S., Depke, R., Engels, G:: Process-Oriented, Flexible Composition of Web Services with UML. Int'l Workshop on Conceptual Modeling Approaches for e-Business: A Web Service Perspective (eCOMO 2002), (2002)

[16]UN/CEFACT, "ebXML - BPSS v1.10", Oct. 2003, http://www.untmg.org

[17]UN/CEFACT, "Core Components Technical Specification V2.01", Nov. 2003, http://www.untmg.org

[18]UN/CEFACT, "UMM Foundation Module 1.0, Draft ", Aug. 2005, http://www.untmg.org