Registering a Business Collaboration Model in Multiple Business Environments

Birgit Hofreiter and Christian Huemer

Institute of Distributed and Multimedia Systems
University of Vienna, Liebiggasse 4, 1010 Vienna, Austria
{birgit.hofreiter,christian.huemer}@univie.ac.at

Abstract. Today business registries are regarded as means of finding services offered by a business partner. However, business registries might also serve as means of seaching inter-organizational business process definitions that are relevant in one's own business environment. Thus, it is important to define in which environments an inter-organizational business process definition is valid. Furthermore, environment-specific adaptations of the business process definition might be registered. In this paper the business process definitions are based on UMM business collaboration models. We discuss two approaches: Firstly, the binding of a model to business environments is specified within the model itself. Secondly, the binding of a model to business environments is defined in the registry meta-data.

1 Motivation

In recent years a lot of interest has been directed toward Web Services in particular and Service-oriented Architectures (SOA) in general. A SOA provides an appropriate framework for inter-organizational systems. An organization provides services and publishes the interfaces of these services in a registry. A potential partner searches the registry for suitable services. To facilitate the search both business partners and its services might be classified according to their business environment. For example, industry classification schemes like NAICS, product classification schemes like UN/SPSC or geographical classification schemes like ISO 3166 are candidates to describe the business environment. It follows that a registry is a core part of a SOA.

The SOA concept assumes that once an organization finds a suitable service in the registry, it is able to bind its own interface to this service. Therefore, it is necessary that the interfaces of both organizations are complementary to each other. Acquiring a product from a business partner usually results in a complex inter-organizational business process. Hence, a binding of a simple, atomic service is not sufficient. Rather, a binding of complex, composite services on each partner's side is necessary. This requires that both the choreography and the data exchanged meet the expectations of the organizations involved. If business partners develop their interfaces in isolation from each other, it is very unlikely that the interfaces will match each other. This hinders interoperability and organizations are not able to do business electronically.

As a consequence organizations must share a common business process in the collaborative space. UN/CEFACT's Modeling Methodology (UMM) [29] provides a

methodology to develop collaborative business process models. These are called business collaboration models in UMM. A business collaboration model exactly describes both the choreography in the collaborative space and the data to be exchanged. Furthermore, the interfaces on each partner's side can be derived from a business collaboration model.

If an organization wants to do business electronically, it has first to select a business collaboration model to participate in. In a next step it has to bind its private interfaces to the collaborative process. This approach corresponds to the one envisioned by ebXML [4]. Accordingly, business collaboration models must be available in a registry. It is important to navigate to a business collaboration model independent from any business partner supporting the model. Usually, an organization knows in which business environment it participates in. Hence, navigation to a business collaboration model should be realized by business environments supporting the model.

Once an organization has discovered a business collaboration model of interest and has bound its private interfaces to the collaborative process, it will register this fact in a registry. This means it is not only able to register its private choreography, but also the implementation of a role in a collaborative process. An organizations simply searches for organizations supporting a complementary role in a conjunct collaborative process in order to find a potential business partner. Thereby, we avoid the hopeless task of finding matching private choreographies.

In this paper we concentrate on the classification of the collaborative business process in a registry. In particular, we present approaches to classify UMM business collaboration models. However, the approaches might be easily adopted to other languages for describing collaborative business processes, like the ebXML business process specification schema (BPSS) [30] or the web services choreography description language (WS-CDL) [10].

The remainder of this paper is structured as follows. In Section 2 we describe related work on collaborative business processes and registries. Since our work is based on UMM we provide a tour through UMM in Section 3. In Section 4 evaluate the current binding of UMM model to business environments. We propose an alternative and present an approach that allows variations for different environments. Section 5 presents an approach to define bindings external to the UMM model in the registry meta-data. It present also a concept to include environment-specific solutions in the registry meta-data. Section 6 summarizes the paper.

2 Related Work

Since it is our goal to register collaborative business processes in a registry, we have a look on related work on registries as well as on collaborative business processes. In the area of collaborative business processes we must distinguish between the analysis & design (i.e. modelling) of collaborative business processes and choreography languages that are consumed by software components to monitor and/or execute a business process. The related work covers both standards and academic research in these areas.

The idea of defining business processes crossing organizational boundaries goes back to ISO's Open-edi reference model [8]. A first implementation of the choreogra-

phy aspects of this model was a Petri-Net approach contributed by Lee [12]. Also other authors used Petri-Nets to define the workflow between organizations [13,15,33]. In addition to the workflow approaches, collaborative processes were also considered from a business transaction perspective. Web Services are seen as an enabler to implement long-running business transactions in a distributed environment of different business partners [16,22].

A lot of different standard languages to describe an orchestration or a choreography of a process have been developed, e.g. Business Process Modeling Language (BPML)[1], Business Process Execution Language (BPEL) [2], Web Services Choreography Interface (WSCI) [35]. An orchestration describes the order in which a composite Web Service invokes other Web Services to realize its function. A choreography describes to the outside world in which order communication with Web Services is expected to fulfill an overall function (c.f. [23]). The specification of collaborative business processes focuses on a complementary choreography between business partners. BPEL is not only the most commonly supported orchestration language, but supports an abstract version for choreographies. Hence, BPEL seems to be the right choice [14]. However, BPEL always describes a choreography from the viewpoint of a single partner. This means it is not possible to describe a single choreography for the overall collaboration. To find a business partner in a registry one cannot refer to a common process, but must perform a complicated match-making of the private process interfaces [36]. As a consequence, the first draft of the Web Services Choreography Description Language (WS-CDL) [10] has been developed to specify a Web Services choreography from a neutral perspective. Within the ebXML framework, the business process specification schema (BPSS) always describes the choreography of a business collaboration from an overall perspective.

Since choreography languages and Web Services are expressed in XML, there have been attempts to model them in a graphical syntax. For this purpose BPMI is developing the Business Process Modeling Notation (BPMN) [34]. This notation presents the amalgamation of best practices in the business process modeling community. Other approaches used UML to visualize Web Services and their choreography [18,24,28]. More advanced approaches provide a development process for inter-organizational business processes. These are either driven by existing private workflows [11] or driven by the inter-organizational requirements instead of the private ones [17]. The latter approach is also used in the development of RosettaNet Partner Interface Processes (PIPs) [25] and UN/CEFACT's modeling methodology (UMM) [29]. In this paper we use UMM to describe and register the collaborative business processes. Thus, we describe UMM in the following section in more detail. In previous publications we have shown a mapping from UMM to BPSS [7] and to BPEL [6], which can then be registered by a similar mechanism as proposed in this paper.

The Web Services regirstry standard is UDDI [20]. The one of ebXML framework is the ebXML Registry and Repository [19]. Both allow organizations to find the services of each other in order to bind them and to do business by the exchange of XML messages. However, the information models of both standards are significantly different. UDDI distinguishes four types of data entries. A *business entity* includes information about an organization. It references a set of *business services* describing its servic-

es. A *business service* includes *binding templates* providing the service access points. A *binding template* refers to a *tModel* representing the technical specification of a service type. The ebXML registry information model (RIM) describes the structure of a *registry entry* entity. This entity contains an ebXML object and a *classification node* entity. The latter is used to create classifications or ontologies for these objects.

In this paper we do not concentrate on the details of how an object, i.e. a business collaboration model, is managed by the registry and its information model. We are more interested in classifying the business objects within the registry. Various authors have shown that extensions to UDDI and ebXML registries in order to enrich the semantic annotation of services [3,21,26,27]. The classification schemas that seem to be most relevant for our aproach is the the context driver concept used in ebXML core components [31] and the UN/CEFACT catalog of common business processes [32].

3 UN/CEFACT's Modeling Methodology (UMM)

UMM is a methodology for describing inter-organizational business processes. It is based on UML. UMM defines a UML profile - i.e. a set of stereotypes, tagged values and constraints - in order to extend the UML meta model for the special purpose of B2B modeling. The UMM methodology leads to so-called business collaboration models. A business collaboration model unambiguously defines the choreography in the collaborative space between the business partners as well as the information exchanged in each collaborative action. It does in no way standardize what is happening in the private space internal to an organization. A UMM business collaboration model focuses on the business logic only. This means it is not specific to any implementation technology, like Web Services, ebXML or UN/EDIFACT. In order to register UMM models the graphical UML syntax must be expressed in a machine-readable format. Currently, UN/CEFACT's business collaboration schema specification (BCSS) project is defining those XML metadata interchange (XMI) flavours that might be used to capture a UMM business collaboration model. Thereby, BCSS makes use of JSR40/ JMI project of the Java Tools Community (JTC) [9].

The UMM methodology comprises four steps that are similar to the first steps in a software development process. In order to demonstrate these steps we use an oversimplified purchase order management example. The resulting artefacts are presented in Fig. 1. The ***business domain view*** is used to gather existing knowledge. The UML concept of use cases and associated worksheets are the tools to describe on a high level the business processes that are of interest to stakeholders. Fig. 1a presents a subset of the business processes that are relevant to the stakeholder *retailer* in our example. We do not depict business processes of interest to any other stakeholder. Furthermore, the business processes are classified into groups to get a better overview of the domain under consideration. This helps to identify possible collaborations in the next step, the ***business requirements view***. Again use cases and associated worksheets are used to collect the requirements of identified business collaborations. It is important that the requirements present a harmonized view of the expectations of all stakeholders. Fig. 1b shows a use case diagram for the identified business collaboration *purchase order manage-*

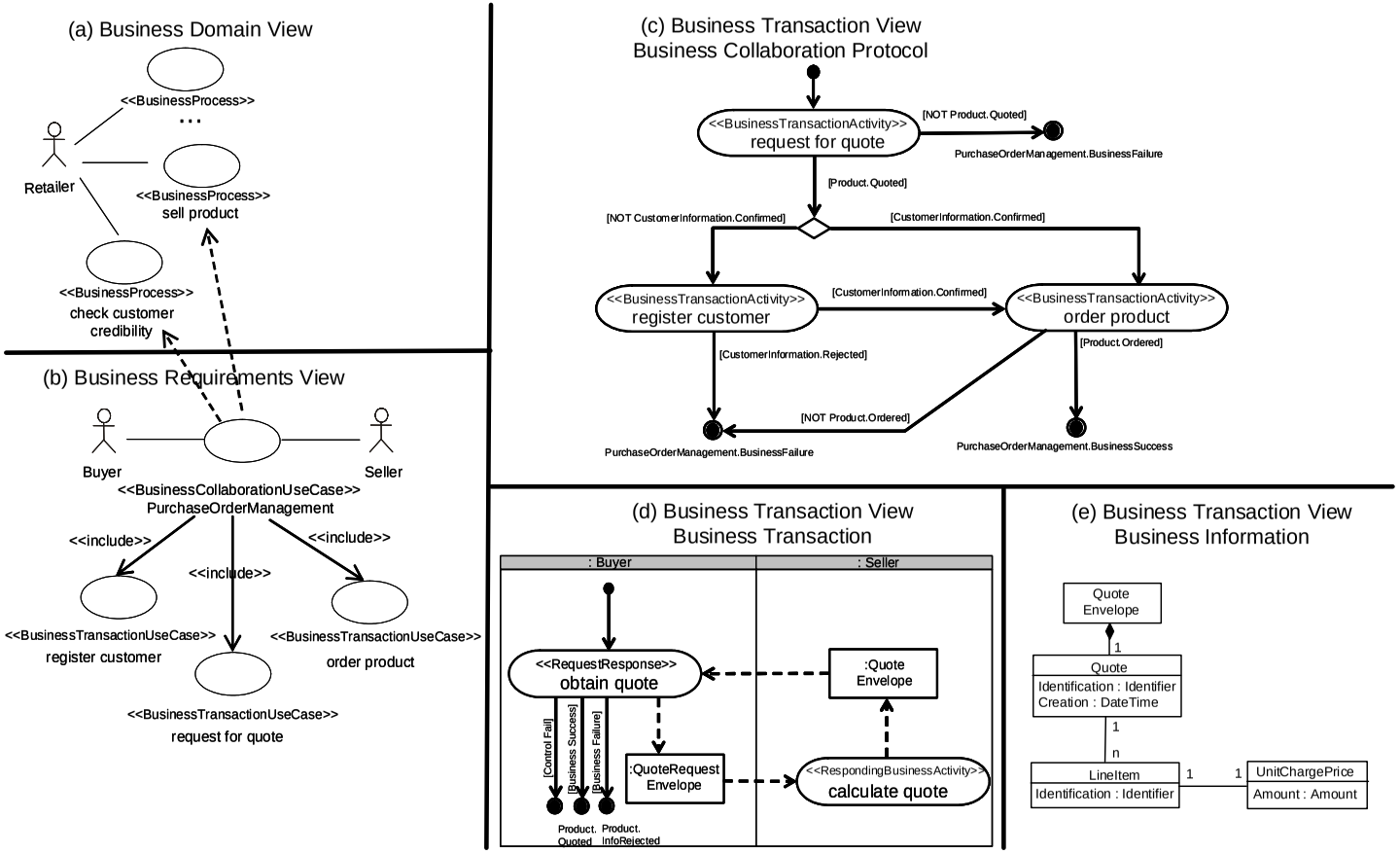

Fig. 1. UMM artefacts for business collaboration: purchase order management

ment. The corresponding business collaboration use case includes dependencies to relevant business processes, as described by stakeholders. Moreover, the business collaboration use case of our purchase order management is built by more basic operations: *register customer*, *request for quote*, and *order product*. This is denoted by the include relationships to the corresponding business transaction use cases.

The ***business transaction view*** covers the analysis model defining the choreography and the information exchanged. It consists of three main artefacts: business collaboration protocol, business transaction and business information. Communication in a business collaboration is about aligning the information systems of the business partners. Aligning the information systems means that all relevant business objects (e.g. purchase orders, line items, etc.) are in the same state in each information system. If a business partner recognizes an event that changes the state of a business object, it initiates a business transaction to synchronize with the collaborating business partner. It follows that a business transaction is an atomic unit that leads to a synchronized state in both information systems. A UMM transaction follows always the same pattern due to its strict definition. The *request for quote* business transaction (Fig. 1d) follows this pattern. 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 Fig. 1d 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. Depending on the decision a product is finally in state *quoted* or *information rejected*. Each of the business transaction use cases of Fig. 1d results in a corresponding business transaction.

The requirements of a business collaboration use case are transformed into the choreography of a so-called business collaboration protocol. Fig. 1c shows the business collaboration protocol for our *purchase order management*. It defines a choreography amongst business transactions, namely *request for quote*, *register customer* and *order product*. It follows that each activity in the business collaboration protocol is refined by the activity graph of a business transaction. For example, the *request for quote* activity in Fig. 1c is refined by the graph of Fig. 1d. The transitions between the activities are guarded by the states of business objects.

Finally, the information exchanged in transactions must be unambiguously defined. Each object in an object flow state is an instance of a class representing an envelope. The aggregates within this envelope are defined in a class diagram. Since UMM is a business state centric approach, the class diagram should only contain information that is needed to accomplish an intended state change. Fig. 1e includes the class diagram for the *quote envelope*, which is exchanged in the business transaction of Fig. 1d. This class diagram is based on ebXML core components [31]. They are assembled and restricted in a way that a business object *product* might be set to state *quoted*, but no additional information is transferred.

The ***business service view*** is the last step of UMM. It describes the design model for the communication between network components implementing the business col-

laboration. The definition includes interaction sequences between two application systems, i.e., protocols of message exchanges, according to the type of business transaction, type of role, security and timing parameters. Since the business service view depends on platform-specific decisions, we do not go into further details.

4 Internal Binding of Models and Environments

4.1 Binding a model to its enviroments

It is envisioned that key players in a business domain will develop UMM business collaboration models. These key players might be standard bodies, like UN/CEFACT itself, industry groups, like EAN/UCC, SWIFT, ad-hoc groups for specific processes, or even market leaders in certain domains. Business collaboration models must be public in order to attract interested parties in the domain. Hence, they are stored in a registry. In this paper, we do not care about the data format used in the registration process - be it a whole model in JMI-compliant XMI [9], or subparts of a model in BPSS [7,30] or WS-CDL [35].

In a next step - similar to the ebXML scenario (c.f. [4], page 8) - an organization is first expected to search the registry for business collaboration models of interest. Large enterprises are expected to implement the local system accordingly, whereas small and medium enterprises might purchase an off-the-shelf software product that supports the business collaboration model. Afterwards the organizations registers the fact that it supports a certain role in the business collaboration model.

Before an organization implements or buys a software it will usually makes sure that the business collaboration is valid in the business domain the organization is 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 must bind it to its business environment. Assume that the model in Fig. 1 was developed by the Austrian Tourism Board for travel packages in Austria. For the sake of re-use, a model should be valid in multiple business environments. Assume that the Cyprus Tourism Board finds the Austrian model in the registry and realizes that this model fits the Cyprus needs as well. There is only a little difference: In Cyprus it is only valid for hotels, not for travel packages.

The current UMM proposal provides bindings of a business process model to a business environment within the model itself. In fact there exist two concepts to realize this binding within a UMM model. First of all, the business domain view package in UMM includes sub-packages for business areas. Each business area includes subpackages for process areas. Business processes and business collaboration use cases will be assigned to a corresponding package. Business areas and process areas should refer to the normative categories specified in the UN/CEFACT common business process catalog [32]. This catalog lists eight business areas: procurement/sales, design, manufacture, logistics, recruitment/training, financial services, regulation, and health care. The five process areas correspond to the phases known from Open-edi [8]: planning, identification, negotiation, actualization, and post-actualization. This is a rather rough classification scheme. It is not granular enough to assign a business collaboration to a real world business environment. The left hand side of Fig. 2 shows a resulting package

structure within the business domain view of a UMM model and the attempt to assign the business collaboration protocol use case to the correct package. Accordingly, the purchase order management collaboration (of our tourism case) is classified into the business area procurement/sales and, within there, in the process area negotiation.

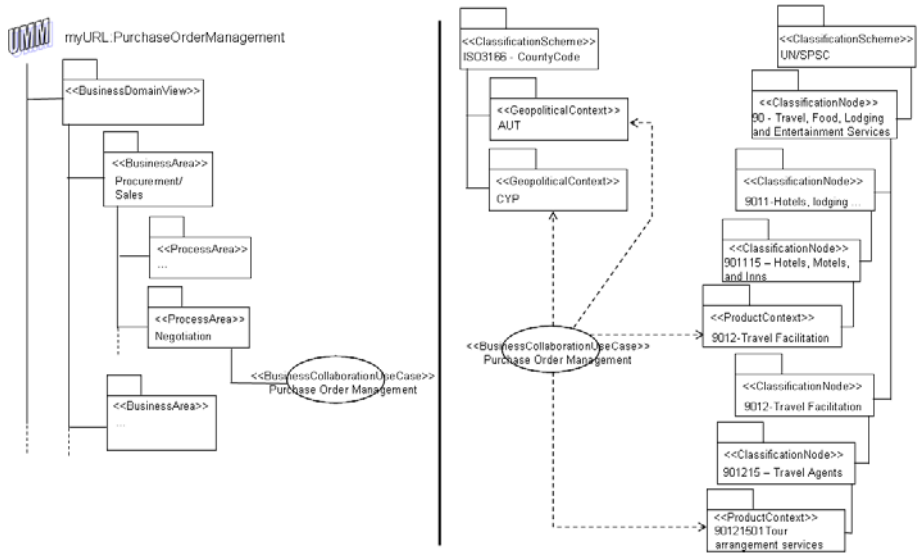

Fig. 2. Binding Business Collaboration Use Cases and Business Environments in UMM

The second classification concept is based on well accepted classification schemes. However, the realization of this concept is deficient. UMM proposes to create a package structure for each classification scheme and its sub-nodes. Dependency relationships from the business collaboration use case to all appropriate classification packages are created. The right hand side of Fig. 2 depicts the classification of our purchase order management according to the UN/SPSC product classification and the ISO 3166 country code. The UN/SPSC uses four levels of classification nodes: segment, family, class, and commodity. In the ISO 3166 only one level of classification exists. The purchase order management depends on the geopolitical contexts of *Austria* and *Cyprus*, and on the product contexts of *tour arrangement services* and *hotels*. By this example two major drawbacks are becoming obvious: Firstly, the package-based approach does not allow category groups. This means the business collaboration is now valid in both countries in both product contexts - which is not correct. Instead one would need a category group for *Austria* and *tour arrangement services* and another group for *Cyprus* and *hotels*. The second drawback is the overwhelming package structure representing the code lists. All the packages are basically empty. They are just created for the sake of establishing dependency relationships.

As a consequence we prefer another mechanism to bind a business collaboration to business environments. We use tagged values instead of packages. This means a tagged value for the business environment is added to the stereotype of a business collaboration use case in the UMM meta model. As a first option, the tagged value refers to an object that is an instance of a class realizing a category group. The other alternative is a string

that follows a predefined format. Since most UML tools do not support tagged values that are instances of a UML element, we selected the second option.

The string could follow the XML structure as used in the category bag of UDDI (c.f. [20] Appendix F). However, we developed our own syntax for the definition of a business environment [5]. This is due to the fact that this definition will also be used in more complicated constraints as outlined further below. The business environment is defined by a business context statement. It is based on the eight context categories defined by ebXML core components to describe a business environment [31]. The business context statement is a boolean expression connecting business context descriptors:

BusinessContextStatement :: =
[<BusinessContext> [<BooleanOperator> <BusinessContextStatement>]? |
[(<BusinessContext> <BooleanOperator> <BusinessContextStatement>)]

BusinessContext ::= <BusinessContextDriver> <relationalOperator> "<literal>"

BusinessContextDriver ::= **`BusinessCollaboration`** | **`BusinessTransaction`** |
`ProductClassification` | **`IndustryClassification`** | **`Geopolitical`** |
`Official Constraints` | **`BusinessProcessRole`** | **`SupportingRole`** |
`SystemCapabilities` | <OtherBusinessContextDriver>

OtherBusinessContextDriver ::= <literal>

BooleanOperator ::= **`AND`** | **`OR`** | **`XOR`**

relationalOperator ::= **`=`** | **`>`** | **`<`** | **`>=`** | **`<=`** | **`<>`**

Accordingly, the business environment tagged value of the purchase order management use case will have the following value:

(Geopolitical = "AUT" AND ProductClassification = "Tour arrangement services") OR
(Geopolitical = "CYP" AND ProductClassification = "Hotels")

4.2 Business environment-specific variations

By now, we are able to define that a business collaboration is applicable in multiple business environments - however only if the collaboration is identically executed in each business environment. A slight variation would require the creation of a new collaboration. Imagine the slight difference in our tourism example: In the Cyprus case the request for quote is nothing else than querying a catalog, whereas in the Austrian case the quote is given on an individual basis and presents a binding offer. This means that the basic choreography for the business transaction and the business collaboration is the same (as depicted in Fig. 1), but response times and security requirements differ. In the Austrian case the response time for receiving the quote is 24 hours and authorization as well as non-repudiation are required. In the Cyprus case the response time is just 1 minute and the security requirements do not apply.

The response time and the security parameters mentioned above are specified in the tagged values of a requesting business activity in a business transaction. Fig. 3 shows all the tagged values of an requesting business activity. Each tagged value carries exactly one value in the original UMM approach. In this case it is impossible to include business context variations. To overcome this situation the tagged value might include an if-statement for the context variations. In this case, the if-clause includes a business context statement as described above and the then-clause sets the value for the corresponding business environment. Further, the if-statement might include elsif-clauses and an else-clause.

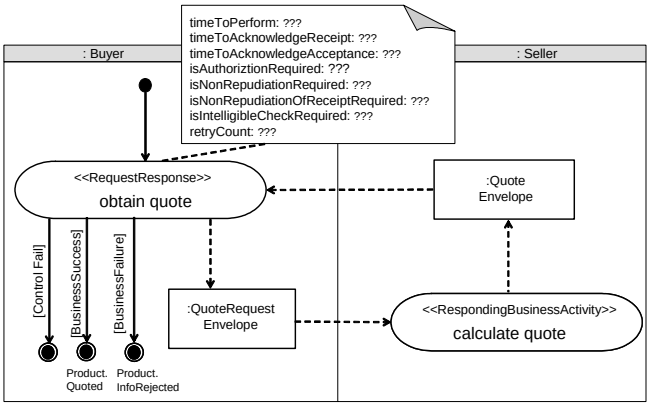

Fig. 3. Tagged Value Variations in a UMM transaction

The tagged value for *time to perform* in our tourism example is specified as stated below. In this statement we assume a default value of *12 hours*, which is neither used in the Austrian nor in the Cyprus case. The boolean values for *is authorization required* and *is non repudiation required* are set by similar if-statements.

if (Geopolitical = "AUT" AND ProductClassification = "Tour arrangement services")
then "PT24H"
elsif (Geopolitical = "CYP" AND ProductClassification = "Hotels")
then "PT1M"
else "PT12H"

Of course, such variations do not only apply to the requesting business activity (cf. [5]). In business transactions, variations exist for the responding business activity and the security parameters of the object flows. In the business collaboration protocol, the tagged values of business transaction activities might differ from environment to environment. Transitions might only be valid in certain environments, or the business states guarding the transition might differ. Finally, the business information exchanged in a transaction might be different. This last variation is tackled by the core components technical specification [31].

5 External Binding of Models and Environments

5.1 Binding a model to its enviroments

In the previous section we defined the binding of a business collaboration model to business environments within the model itself. Thus we call it an internal binding. In this section we concentrate on an external binding. The definition of the applicable business environments are not defined within the model itself, but in the registry. Similarly the differences between business environments must be handled by registry meta-data and not within the model.

The motivation for an external binding is best demonstrated by the simple example we used already before. Imagine the Austrian tourism board developed its purchase order management for travel packages. It registers its model. The Cyprus tourism board detects this model in the registry and finds it appropriate for their hotel domain. In the

case of an internal binding, the Cyprus tourism board must change the model in order to declare it as valid in their environment and to define the context variations. This means a new version of a business collaboration model is created whenever a new domain is interested in it - even if the basics of the model do not change. If the Austrian board wants to update the model later, this might have consequences on versions created by other organizations. This means a complex version mechanism is necessary. Maintenance would be much simpler in the case of external bindings: A business collaboration model is defined independently of the business environment. Metaphorically, each domain organization that accepts the model for their environment puts a sticker on the model. The Austrian tourism board creates the model and puts a sticker on it. Later on, the Cyprus board finds the model and puts another sticker with the Cyprus-specifics on it. If the Austrian Tourism board updates the model it creates a new version, and moves the sticker from the old to the new version. It is up to the Cyprus board to leave the sticker on the old version or to move it to the new one as well.

Of course we do not have any stickers in the electronic world. Instead of stickers, this concept must be realized by the registry meta-data assigned to a business collaboration model. All four main UDDI data structure types provide a structure to support attaching categories [20]. One of these data types is the *tModel*. *tModels* are used to engender the notion of shared specifications. Inasmuch a *tModel* contains the address where a business collaboration model can be found. The *tModel* structure allows to attach a *category bag*. A *category bag* might include *keyed reference groups*. Each group is used to describe a business environment. The following *tModel* is used to bind our purchase order management to Austrian travel packages and Cyprus hotels:

```
<tModel tModelKey="uddi:whoever:umm:purchaseordermanagement">
    <name>http://www.whoever.org/purchaseOrderManagement</name>
    <categoryBag>
        <keyedReferenceGroup>
            <keyedReference
                tModelKey="uddi:uddi.org:ubr:categorization:unspsc"
                keyName="UNSPSC:Tour arrangement services"
                keyValue="90.12.15.01"/>
            keyedReference
                tModelKey="uddi:uddi.org:ubr:categorization:iso3166"
                keyName="GEO:Austria"
                keyValue="AT"/>
        </keyedReferenceGroup>
        <keyedReferenceGroup>
            <keyedReference
                tModelKey="uddi:uddi.org:ubr:categorization:unspsc"
                keyName="UNSPSC:Hotels"
                keyValue="90.11.15.01"/>
            keyedReference
                tModelKey="uddi:uddi.org:ubr:categorization:iso3166"
                keyName="GEO:Cyprus"
                keyValue="CY"/>
        </keyedReferenceGroup>
    </categoryBag>
</tModel>
```

Similarly to UDDI, ebXML includes a mechanism to classify a registry object by the value of an external classification schema [19]. Since an ebXML registry does not care about what is registered, a business collaboration model is a potential registry object. The ebXML mechanisms of classifying classifications allows the grouping of different business environment parameters. Due to space limitation we do not show the details of the resulting ebXML code.

5.2 Business environment-specific variations

So far, we are able to bind a business collaboration model and business environments in the registry meta-data, if the business collaboration model is identical in each environment. Next we have to consider variations in the execution of business process models. We again use the example of a different response time and security parameters in the *request for quote* business transaction to illustrate the approach.

We are facing now the problem that the concepts that may vary - tagged values, transitions, guards - are part of the UMM model itself. Nevertheless we want to specify variations to these concepts without changing the model itself. Hence, a UMM model includes the defaults for these concepts. In our example, the Austrian tourism board registers the model with the default values. For example, the tagged value of *time to perform* of *obtain quote* is *24 hours*. Later the Cyprus tourism board wants to define that the respective *time to perform* is only *1 minute* in their scenario. However, it cannot change the tagged value in the model itself. For this purpose we develop an XML schema to define business collaboration variations. The code below shows an extract of the schema that we need in our example. The root element *business collaboration variation* uses an *id* attribute to reference the original business collaboration model. The elements beneath are used to specify the variations within this model.

```
<element name="BusinessCollaborationVariation">
    <complexType>
        <sequence>
            <element ref="BusinessCollaborationCharacteristics" minOccurs="1" maxOccurs="unbounded"/>
        </sequence>
        <attribute name="baseBusinessCollaborationModelId" type="anyURI"/>
        <attribute name="BaseBusinessCollaborationModelName" type="string"/>
</element>
<element name="BusinessCollaborationCharacteristics">
    <complexType>
        <sequence>
            <element ref="BusinessTransactionActivityCharacteristics"minOccurs="0"maxOccurs="unbounded"/
            <element ref="TransitionCharacteristics" minOccurs="0" maxOccurs="unbounded"/>
            <element ref="BusinessTransactionCharacteristics" minOccurs="0" maxOccurs="unbounded"/>
        </sequence>
        <attribute name="nameId" type="anyURI"/>
        <attribute name="name" type="string"/>
    </complexType>
</element>
```

```
<element name="BusinessTransactionCharacteristics">
    <complexType>
        <sequence>
            <element ref="RequestingBusinessActivityCharacteristics" minOccurs="0"/>
            <element ref="RespondingBusinessActivityCharacteristics" minOccurs="0"/>
            <element ref="RequestingBusinessInformationCharacteristics" minOccurs="0"/>
            <element ref="RespondingBusinessInformationCharacteristics" minOccurs="0"/>
        </sequence>
        <attribute name="nameId" type="anyURI"/>
        <attribute name="name" type="string"/>
    </complexType>
</element>
<element name="RequestingBusinessActivityCharacteristics">
    <complexType>
        <attribute name="timeToPerform" type="duration"/>
        <attribute name="timeToAcknowledgeReceipt" type="duration"/>
        <attribute name="timeToAcknowledgeAcceptance" type="duration"/>
        <attribute name="isAuthorizationRequired" type="boolean"/>
        <attribute name="isNonRepudiationRequired" type="boolean"/>
        <attribute name="isNonRepudiationReceiptRequired" type="boolean"/>
        <attribute name="isIntelligibleCheckRequired" type="boolean"/>
        <attribute name="retryCount" type="integer"/>
    </complexType>
</element>
```

The Cyprus tourism board creates the following instance of this XML schema. The *id* of the business collaboration variation points to the original business collaboration model. It redefines the characteristics of the business collaboration *purchase order management*. Within this business collaboration it redefines the characteristics of the business transaction *request for quote*. In this transaction it changes the tagged values for *time to perform*, *is authorization required* and *is non repudiation required* of the requesting business activity. It is clear that this is the *obtain quote* activity, because a business transaction includes always exactly one requesting business activity.

```
<BusinessCollaborationVariation
    baseBusinessCollaborationModelId="http://www.whoever.org/purchaseOrderManagement">
    <BusinessCollaborationCharacteristics name="purchaseOrderManagement">
        <BusinessTransactionCharacteristics name="requestForQuote">
            <RequestingBusinessActivityCharacteristics
                timeToPerform="PT1M" isAuthorizationRequired="false"
                isNonRepudiationRequired="false"/>
        </BusinessTransactionCharacteristics>
    </BusinessCollaborationCharacteristics>
</BusinessCollaborationVariation>
```

Once the Cyprus board has created this business collaboration activity, it is able to create a *tModel* that refers to this variation. This means the Cyprus board does not add another keyed reference group to the *tModel* of the original model. Instead it creates another *tModel* for the XML Schema of the variation which links to the original business collaboration. The *tModel* of the variation includes a category bag that includes the classification of the Cyprus case. If afterwards another Tourism board comes along, that

wants to use the classification exactly as the Cyprus board specified, it adds its classification to the category bag - which must then include *keyed reference groups* again.

```
<tModel tModelKey="uddi:someoneelse:umm:purchaseordermanagementvariation">
    <name>http://www.someoneelse.org/purchaseOrderManagementVariation</name>
    <categoryBag>
        <keyedReference
            tModelKey="uddi:uddi.org:ubr:categorization:unspsc"
            keyName="UNSPSC:Hotel"
            keyValue="90.11.15.01"/>
        keyedReference
            tModelKey="uddi:uddi.org:ubr:categorization:iso3166"
            keyName="GEO:Cyprus"
            keyValue="CY"/>
        </keyedReferenceGroup>
    </categoryBag>
</tModel>
```

We already mentioned that an ebXML registry does not care about what an registry object is about. As a consequence, the XML file of the Cyprus variation might be a valid registry object, which can be classified accordingly. Again we do not detail the code for reasons of space limitations.

6 Summary

Inter-organizational business processes are usually quite complex. The acquisition of a product from a business partner is not a one step process. Usually a lot of communications between the business partners are necessary. Electronic communication between the applications of the business partners requires an unambiguous choreography and unambiguous information exchanged. If each organizations develops their interfaces independently from each other, it is rather unlikely that they will inter-operate.

Hence there is a need for shared business collaboration models, which exactly define each roles behavior and to which each partner can bind its private processes. UMM provides a methodology to develop these business collaboration models. These business collaboration models must be publicly available. Therefore, this paper deals with the registration of UMM models. We focus on the registration of the models independent of the support by business partners. This means, we do not concentrate on the binding of organizations to roles in business collaboration models.

Organizations will search a registry to find an appropriate model. A search will look for models that are valid in a business environment of interest. Models must be bound to specific business environments. Thus, we had a look on the binding of a model to one or more business environments. Although different business environments share a common model, the execution might be slightly different. So we focused also on the registration of these slight variations.

Both the binding and the variations might be specified either in the UMM model itself or in the registry meta-data. In the current UMM approach a binding is defined within the model itself. An evaluation of the current approach showed that it is insufficient because it does not support category groups and it results in an overloaded package structure. We developed an alternative approach based on tagged values. The cur-

rent UMM does not allow the definition of slight variations. Thus we extended the UMM to allow variations for tagged values, transitions and transition guards.

In contrary to the current UMM vision of defining a binding within the model itself, we investigated in a binding of the model and its environment externally in the registry meta-data. We prefer this approach because the maintenance of the bindings does not effect the models. This results in a much simpler versioning mechanism. We demonstrated by the means of UDDI that this binding mechanism as available of today. ebXML registries would be prepared as well. More complicated is the external definition of variations. For this purpose we had to develop an XML schema that references the original model and captures the variations. Furthermore, we demonstrated its usage within a registry by the means of UDDI again.

This paper was motivated by discussions within UN/CEFACT's TMG group, which is responsible for maintaining the UMM. By this paper we provide a first evaluation of approaches on registering UMM business collaboration models. After discussion of the suggested approaches with TMG, we consider a prototype implementation.

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. Dogac, A., Kabak, Y., Laleci, G.: Enriching ebXML Registries with OWL Ontologies for Efficient Service Discovery. Proc. of RIDE Workshop 2004, IEEE Computer Society (2004)
4. Eisenberg, B., Nickull, D.: ebXML Technical Architecture Specification, v1.0.4. (2001) http://www.ebxml.org/specs/ebTA.pdf
5. Hofreiter, B., Huemer, C., Winiwarter, W.: OCL-Constraints for UMM Business Collaborations. Proc. of 5th Int'l Conf. on Electronic Commerce and Web Technologies (EC-Web 2004), Springer LNCS, (2004)
6. Hofreiter, B., Huemer C.: Transforming UMM Business Collaboration Models to BPEL. Proc. of OTM Workshops w004. Springer LNCS, Vol. 3292, (2004) 507-519
7. Hofreiter B., Huemer, C., Kim J.-H: Choreography of ebXML business collaborations. accepted and to appear in: Journal of Information Systems and e-Business. Springer (2005)
8. ISO: Open-edi Reference Model. ISO/IEC JTC 1/SC30 ISO Standard 14662 (1997)
9. Java Community Process: JSR-000040 The JavaTM Metadata Interface (JMI) Specification. http://jcp.org/aboutJava/communityprocess/final/jsr040/index.html
10. Kavantzas, N. et al: Web Services Choreography Description Language, Version 1.0. W3C (2004) http://www.w3.org/TR/ws-cdl-10
11. J.-Y. Jung, W. Hur, S.-H. Kang, H. Kim: Business Process Choreography for B2B Collaboration. IEEE Internet Computing, (2004)
12. R.M. Lee: Documentary Petri Nets: A Modeling Representation for Electronic Trade Procedures. In: Business Process Management, Models, Techniques, and Empirical Studies. Springer LNCS, Vol. 1806 (2000) 259 - 375
13. Lenz, K., Oberweis, A.: Interorganizational Business Process Management with XML Nets. In: Advances in Petri Nets. Springer LNCS, Vol. 2472 (2003)
14. Leymann, F., Roller, D., Schmidt, M.-T.: Web Services and Business Process Management. IBM Systems Journal, Vol. 41, No. 2, 2002

15. Ling, S., Loke S.W.: Advanced Petri Nets for Modelling Mobile Agent Enabled Interorganizational Workflows. Proc. of 9th IEEE Int'l Conf. and Workshop on the Engineering of Computer-Based Systems (ECBS 2002), IEEE Computer Society (2002)
16. Little, M.: Transactions and Web Services. Communications of the ACM, Vol. 46, No. 10 (2003) 49 - 54
17. 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.
18. Mantell, K.: From UML to BPEL - Model Driven Architecture in a Web Services World. IBM Developer Works (2003), http://www-128.ibm.com/developerworks/webservices/library/ws-uml2bpel/
19. OASIS: ebXML Registry Information Model v2.0. (2001) http://www.ebxml.org/specs/ebrim2.pdf
20. OASIS: UDDI Version 3.0.2. http://uddi.org/pubs/uddi_v3.htm
21. Overhage, S., Thomas, P.: WS-Specification: Specifying Web Services Using UDDI Improvements. Web, Web-Services, and Database Systems: NODe (2002)
22. Papapzoglou, M.P.: Web Services and Business Transactions. WWW, Internet and Web Information Systems, Vol. 6, Kluwer (2003) 49-91
23. Peltz, C.: Web Services Orchestration and Choreography. IEEE Computer, Vol. 36, No. 10, (2003) 46-52
24. Provost, W.: UML for Web Services.XML.com (2003) http://www.xml.com/lpt/a/ws/2003/08/05/uml.html
25. RosettaNet: RosettaNet Implementation Framework, Core Specification V02.00.01. (2002) http://www.rosettanet.org/rnif
26. Shaikkhali, A., Rana, O.F., Al-Ali, R., Walker, D.W.: UDDIe: An extended registry for web services. Proc. of the 2003 Symposium on Applications and the Internet Workshops (SAINT'03) (2003)
27. Srinivasan, N., Paolucci, M., Sycara, K.: Adding OWL-S to UDDI, implementation and throughput. 1st Int'l Workshop on Semantic Web Services and Web Process Composition (SWSWPC) (2004)
28. 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)
29. UN/CEFACT: UMM Meta Model, Revision 12; (2003) http://www.untmg.org/downloads/General/approved/UMM-MM-V20030117.zip
30. UN/CEFACT: ebXML - Business Process Specification Schema v1.10. (2003) http://www.untmg.org/downloads/General/approved/ebBPSS-v1pt10.zip
31. UN/CEFACT: Core Components Technical Specification, Version 2.01. (2003) http://www.unece.org/cefact/ebxml/CCTS_V2-01_Final.pdf
32. UN/CEFACT: Common Business Process Catalog, v2.0 (2005) Not published yet (link will be included in final version)
33. van der Aalst, W.M.P.: Interorganizational Workflows: An Approach based on Message Sequence Charts and Petri Nets. Systems Analysis - Modelling - Simulation, Vol. 34, No. 3 (1999) 335-367
34. White, S: Business Process Modeling Notation Working Draft , V 1.0. (2003) http://www.bpmi.org/bpmn-spec.esp
35. W3C: Web Service Choreography Interface V 1.0. (2002) http://www.w3.org/TR/wsci/
36. Wombacher, A., Fankhauser, P., Mahleko, B., Neuhold, E.: Matchmaking for Business Processes Based on Choreographies. Int. J. of Web Services Research, Vol. 1, No. 4 (2004)