Journal on Information Systems and e-Business manuscript No.
(will be inserted by the editor)

Extending UN/CEFACT's Modeling Methodology by a UML Profile for Local Choreographies

Birgit Hofreiter

Received: 30 July 2007 / Accepted: 25 October 2007

Abstract UN/CEFACTs Modeling Methodology (UMM) is a UML profile for specifying global choreographies of inter-organizational e-business systems. As we outline in this paper, the practical use of UMM is limited to bi-lateral business collaborations, since it does not support nested business transactions. This means UMM does not support multi-party business collaborations. UN/CEFACT argues that UMM serves as model capturing the agreements and commitments between business partners. These agreements and commitments are always on a bi-lateral basis. However, a business partner in the middle of a supply chain must establish multiple agreements and commitments with multiple partners. It is the local choreography of a business partner that binds the various bi-lateral models leading to a multi-party choreography. Unfortunately, UN/CEFACT does not give any guidance on how to model the local choreographies. We close this gap by extending UMM by a UML profile for local choreographies.

Keywords Business Process Modeling · Inter-Organizational Systems · Conceptual Modeling · Standards · Unified Modeling Language

1 Introduction

The idea of exchanging business data between the information systems of collaborating business partners exists for quite a while. It became known as electronic data interchange (EDI) (Schatz, 1988). For a long time standardization efforts concentrated on the development of standard business document types. However, a common understanding of the data exchanged is just one part of the solution. Inter-organizational systems require interlacing the business processes of the participating partners.

Birgit Hofreiter
Department of Distributed and Multimedia Systems, University of Vienna, Liebiggasse 4/3-4, 1010 Vienna, Austria
Tel.: +43-1-4277-39634
Fax: +43-1-4277-39649
E-mail: birgit.hofreiter@univie.ac.at
Present address: Department of Information Systems, University of Technology Sydney, PO Box 123, Broadway NSW 2007, Australia

More recently, we see efforts describing the orchestration and/or choreography of business processes. Orchestration and choreography describe closely related, but well distinguished concepts (Barros *et al.*, 2005), (Peltz, 2003). Orchestration deals with the sequence and conditions in which one business process calls its components to realize a business goal. Choreography describes business processes in a peer-to-peer collaboration. It 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 partners 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. A global choreography has the potential to achieve an agreement between the partners. Local choreographies enable the configuration of each partners system.

In order to establish a collaboration between business partners we see two main approaches. Firstly, one business partner announces his local choreography. All other business partners that want to collaborate must adopt their interfaces accordingly. Secondly, business partners start agreeing on a common global choreography - which may be based on a predefined reference model specified by standards organizations or industry consortia. Next, each partner binds its own local choreography to the public one.

The United Nations Centre for Trade Facilitation and E-Business (UN/CEFACT) is a standards organization that has committed itself to develop such predefined reference models. UN/CEFACT became known by creating and maintaining the UN/EDIFACT standard. Since UN/CEFACTs mission is rather to develop trade procedures than to develop IT-platform specific solutions, it started to standardize business scenarios independent of the IT platform: Core Components are platform-independent business document building blocks (Stitzer and Crawford, 2003). UN/CEFACTs Modeling Methodology (UMM) models the choreography and data exchange commitments independent of the IT.

Inasmuch, the UMM model becomes a kind of contract that guides the choreography. Since most commitments are made between two partners only, a UMM model - such as most contracts - is agreed upon between two parties. Also similar to a contract, a UMM model describes the commitments on the information flow from a neutral perspective. It follows that UMM describes a bi-lateral and global choreography. The UML profile of the UMM foundation module (Dietrich *et al.*, 2006) - which was edited by our Vienna team - has been defined exactly for this special purpose.

This means that UMM is designed neither to model local choreographies nor to model multi-party choreographies. This means UMM does not allow to model a supply chain which is built by three or more partners. Since UMM is designed to model agreements and commitments between partners, it is argued that a supply chain is built by multiple bi-lateral agreements and commitments. It is said to be a local decision of an individual partner to select his predecessors and successor in the supply chain. Nevertheless, it is not possible to use UMM for modeling a partners local choreography showing how he interacts with predecessors and successors.

In this paper we present an approach how to model local and multi-party choreographies. It should be noted that in this paper the terms business partner, party, and organization refer to economically independent units, which are not necessarily legally independent. In other words, our approach may also be used in an evironment where two or more economically independent departments of the same company collaborate. Since it is important that the local and multi-party choreography of a certain part-

ner respects the commitments defined in the global and binary choreographies of the UMM models, we concentrate on bridging global and local choreographies. As said before, UMM is defined as a UML profile. In order to build a bridge our local, multi-party choreography is defined as a UML profile as well, that specifies clear extension points to UMM and includes constraints between stereotypes of both profiles. Our approach also provides a means for connecting multiple bi-lateral choreographies leading to multi-party choreographies.

The remainder of this paper is structured as follows: We provide an overview of related work in section 2. In section 3 we first introduce UMM by the means of a simple example. In section 4 we demonstrate UMMs limitations to model multi-party choreographies. Section 5 introduces our approach to extend the UMM by local, multi-party choreographies. Again we start off with a simple example. Next we define the stereotypes and constraints of our UML profile. A summary in section 6 concludes the paper.

2 Related Work

The idea of defining business processes crossing organizational boundaries goes back to ISOs Open-edi reference model (ISO, 1995). A first implementation of the choreography aspects of this model was a Petri-Net approach contributed by Lee (Lee, 2000). Also other authors used Petri-Nets to define the workflow between organizations (Lenz and Oberweis, 2003),(Ling and Loke, 2002),(van der Aalst, 1999). In addition to the Petri-Nets formalism, conceptual modeling languages became popular for describing inter-organizational processes for the purpose of understanding and communication. The two most significant techniques for conceptual modeling of business processes are the Unified Modeling Language (UML) (Booch *et al.*, 2005) and the Business Process Modeling Notation (BPMN) (White, 2006). The main advantage of these techniques is the use of intuitive notations yet capable to represent complex process semantics and interactions. UML is a general purpose modeling language. In order to use UML for modeling business processes different authors have developed either just guidelines or a UML profile that customizes UML for business process modeling by a set of stereotypes, tag definitions and constraints on the UML meta model. Customizations of UML for modeling business processes internal to a company are described in (Korherr and List, 2006),(List and Korherr, 2005),(Penker and Eriksson, 2000). Beside UMM, UML customizations for modeling inter-organizational processes are described in the RosettaNet Framework (Kraemer and Yendluri, 2002) and in Kramler et al. (Kramler *et al.*, 2005). It should be noted that in 2000 the company EDIFECS contributed the RosettaNet methodology to UN/CEFACT which merged it into UMM. Inasmuch, the UMM concept of business transactions - described in the next section - is identical to the one for RosettaNet PIPs. Thus, the criticism on UMM described in section 4 also applies to Rosetta Net.

Another popular way of describing (inter-organizational) business processes was triggered by the growing importance of XML and Web Services. Different text based process modeling languages appeared. These usually had no graphical notation, but may be interpreted by software allowing the tracking or even execution of the business process. The Business Process Execution Language for Web Services (BPEL4WS) (Andrews *et al.*, 2003),(Leymann *et al.*, 2002) became the most popular language in this area. It is able to describe the orchestration of executable business processes, but also

the message exchanges in a local choreography. The transformation of a global choreography to BPEL processes have been described by Khalaf for RosettaNet (Khalaf, 2007) and by ourselves for UMM (Hofreiter *et al.*, 2007 a). In the area of Web Services, the Web Services Choreography Description Language (WS-CDL) (Kavantzas *et al.*, 2005) is the choice for modeling global choreographies. However, WS-CDL uses its own set of control flow constructs which are hard to map to those of BPEL. In order to overcome this limitation, BPEL4chor (Decker *et al.*, 2007) has recently been proposed to extend BPEL for describing global choreographies. Another XML-based language for describing global choreographies is ebXMLs Business Process Specification Schema (BPSS) (UN/CEFACT, 2003). Since the BPSS is based on UMM and is more or less an XML representation of UMMs business transaction view (Hofreiter *et al.*, 2006 a), it inherits the limitations in modeling multi-party collaborations as described in section 4. OASIS BPSS successor called ebBP 2.0 has recognized these limitations and provides a workaround called *complex business transaction activity* that works in simple scenarios.

A UMM model - as explained in the following section - may contain multiple business collaborations including many different business transactions. It has all business collaborations and business transactions relevant to a business goal under consideration. However, this does not mean that all business collaborations and business transactions must originate from that model. They may have been originally developed in another UMM model, exported and published in a registry. A UMM model may import and re-use business collaborations and business transactions stored in the registry. We describe an approach to manage UMM models in an ebXML registry in (Hofreiter *et al.*, 2006 c).

There exist some approaches that take care of the relationship between orchestration, local choreography, and global choreography. In particular, the different perspectives can be transformed into each other, that is, the global choreography can be transformed into the local choreography, and the local choreography can be transformed into an orchestration. The basic idea of transforming a global choreography to a local choreography is partitioning the global choreography in a way that the messages used in the resulting local choreographies are matched to the corresponding party. This partitioning is achieved by eliminating messages which are not related to the particular party. Such an approach is described by Aalst in (van der Aalst and Weske, 2001). The transformation of a local choreography to a global choreography can be realized by relating message exchanges of different local choreographies that semantically complement one another. An approach based on workflow nets is proposed by Aalst (van der Aalst, 2002),(van der Aalst, 1999). Piccinelli et al.(Piccinelli *et al.*, 2002) propose another transformation approach in particular for peer-to-peer collaborations in Web Services environments.

These approaches differ significantly from our proposed approach. Evidently, they are based on Petri Nets and Web Services, respectively, whereas our approach extends the UML. However more importantly, those approach deal with the automatic transformation between orchestration, local choreography, and global choreography and consistency checking between the different perspectives. Our approach does not deal with any kind of automatic transformation. Transformations from multiple global choreographies into a single local choreography cannot be achieved automatically and must be done by hand. Our approach provides a mechanism to specify the dependencies between global and local choreographies. These dependencies may be used in a reasoning approach to check consistency following the ideas of Aalst et al.

3 UN/CEFACTs Modeling Methodology (UMM)

As already outlined in the motivation, our approach is based on the UN/CEFACT modeling methodology (UMM), which was edited by our research group. In this section we introduce those UMM concepts that are relevant for understanding our approach. The reader interested in more UMM details is referred to the technical specification (Dietrich *et al.*, 2006) or our publication in (Hofreiter *et al.*, 2006 b). The UMM meta model is defined as a UML profile. Accordingly, it is built on top of the UML meta model. This profile defines a set of stereotypes including tagged definitions that are specific for modeling business collaborations in B2B. Since UMM is based on the UML meta model, the relationships between the stereotypes are defined by OCL constraints on the UML meta model. The constraints as specified in the specification lead to very restrictive UMM business collaboration models. For example, these constraints mandate that a UMM model includes three top level packages each stereotyped according to one of the three main views: the *business domain view* (BDV), the *business requirements view* (BRV), and the *business transaction view* (BTV). Similarly, each of the main views has to include a well-defined number of stereotyped sub-views. The stereotyped modeling elements within each subview and their relationships are also well defined.

Due to page limitations we are not able to detail all the stereotypes, tag definitions and OCL constraints. For a better understanding of the UMM stereotypes and their inter-dependencies we walk through the UMM by the means of a simplified, but still realistic example of an `order from quote` in `purchase management`. The relevant artifacts of our example are depicted in figure 1. On the top of this figure we see the structure of our `purchase management` model.

The first top level package is stereotypes as *business domain view* (BDV). The BDV is used to gather existing knowledge from stakeholders and business domain experts. It is the goal to get a basic understanding of the business processes of the business partner under consideration. Since our approach is interlinked with the subsequent UMM views, we do not detail the BDV artifacts. We just show the *business partner types* `retailer` and `wholesaler` which are used subsequently.

The second top level package is stereotyped as *business requirements view* (BRV) specifies the requirements for the business collaboration under development. We omit to detail the *business process view* and the *business entity view*. A *transaction requirements view* defines the *business transaction use case* for a certain task and binds the two *authorized roles* involved. The *authorized roles* are defined in the exact context of the *business transaction use case*. In our example we have two *transaction requirements views*: `request for quote` (2) and `place order` (3). The *authorized roles* in `place order` are `order requestor` and `order responder` whereas the *authorized roles* in `request for quote` are `quote requestor` and `quote responder`.

The *collaboration requirements view* includes a *business collaboration use case*. The *business collaboration use case* aggregates *business transaction use cases* or recursively structured *business collaboration use cases*. This is manifested by *include* associations. In our example the *business collaboration use case* `order from quote` (1) includes the *business transaction use cases* `request for quote` (2) and `place order` (3). Furthermore, the *authorized roles* participating in the *business collaboration use case* must be defined within the context and namespace of the *collaboration requirements view*. The same is valid for the *authorized roles* participating in the *business transaction use cases*. *Maps to* dependencies are used to define which *authorized role* of a *business collaboration*

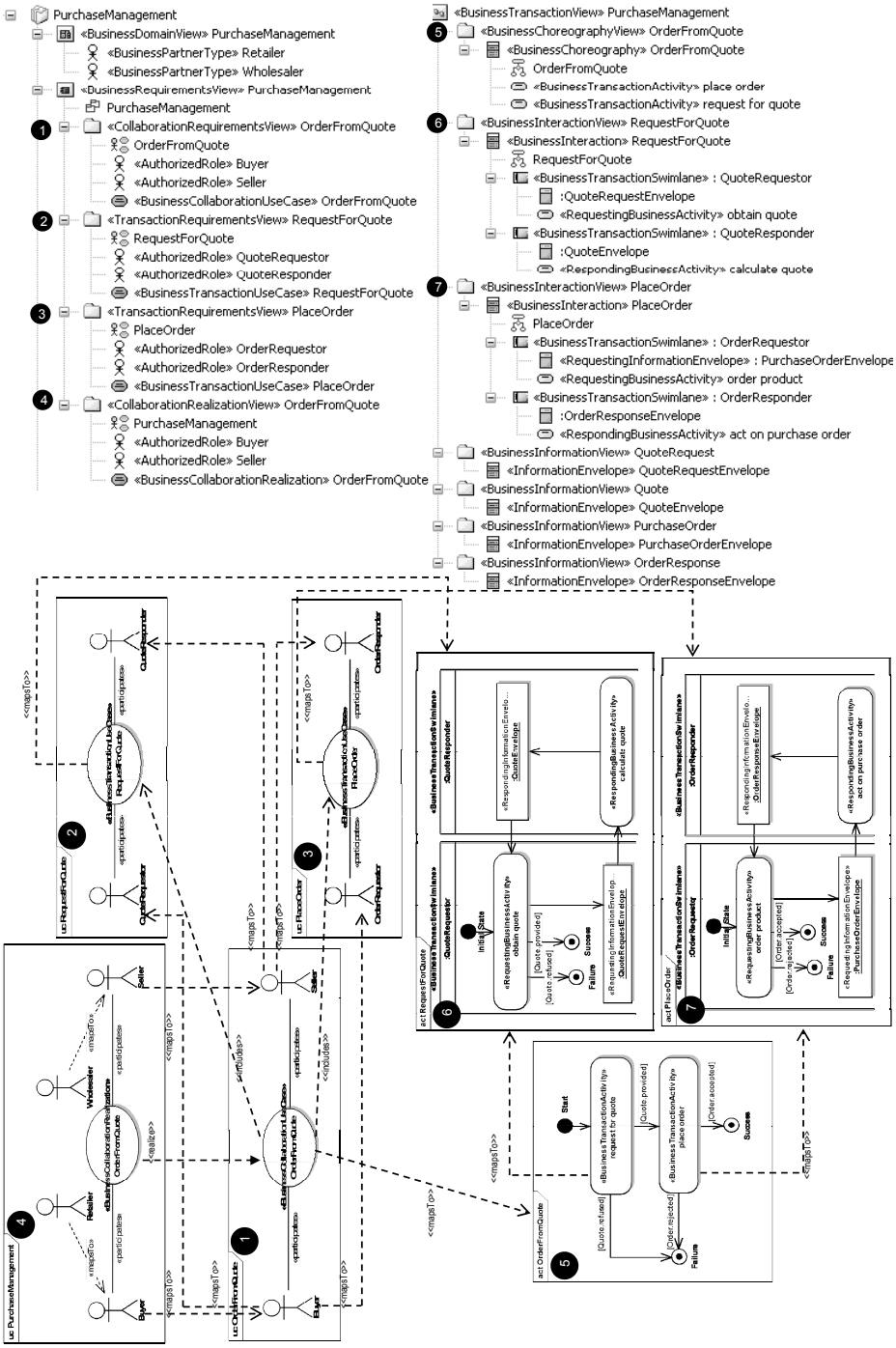

Fig. 1 UMM by Example: Purchase Management

use case plays which role in an included *business transaction use case* (or recursive *business collaboration use case*). In our example the `buyer` of `order from quote` (1) plays the `quote requestor` in `request for quote` (2) and the `order requestor` in `place order` (3). The `seller` of `order from quote` (1) plays the `quote responder` in `request quote` (2), and the `order responder` in `place order` (3).

The *collaboration realization view* covers the *business collaboration realization* - which is a kind of use case that does not elaborate any new requirements. A *business collaboration realization* realizes a *business collaboration use case* between a specific set of *business partner types*. This is indicated by a *realize* association. The *business collaboration realization* `order from quote` (4) realizes the *business collaboration use case* `order from quote` (1). The *business partner types* participating in the *business collaboration realization* are the ones already defined in the BDV and, thus, are not re-defined in the namespace of the *collaboration realization view*. A *maps to* dependency defines which participant of a *business collaboration realizatio*n plays which role in the *business collaboration use case*. In the `order from quote` realization (4) the `retailer` plays the `buyer` and the `wholesaler` acts as `seller`.

Finally, the third top level package captures the artifacts of the *business transaction view* (BTV). The BTV builds upon the BRV and defines a global choreography of information exchanges and the document structure of these exchanges. The choreography described in the requirements of a *business transaction use case* is represented in exactly one activity graph of a *business transaction*. A *maps to* dependency between them allows traceability between the requirements and the *business transaction*, which is defined in a *business interaction view*. In our example, the `request for quote` requirements (2) are mapped to a corresponding choreography (6). The same mapping is made for the `place order` requirements (4) to its choreography (7).

A *business transaction* is characterized as follows: If an *authorized role* recognizes an event that changes the state of a *business entity*, it initiates a *business transaction* to synchronize with the collaborating *authorized role*. A *business transaction* is an atomic unit that leads to a synchronized state in both information systems. We distinguish one-way and two-way *business transactions*: In the former case, the initiating *authorized role* reports an already effective and irreversible state change that the reacting *authorized role* has to accept. In the other case, the initiating role sets the *business entity/ies* into an interim state and the final state is decided by the reacting *authorized role*. It is a two-way transaction, because business information flows from the initiator to the responder to set the interim state and backwards to set the final and irreversible state change. Irreversible means that returning to an original state requires compensation by another *business transaction*.

UMM distinguishes two types of one-way transactions. If the business information sent is a formal non-repudiatable notification, the transaction is called *notification*. Otherwise the transaction is known as *information distribution*. Furthermore, there exist four different types of two-way transactions. If the responder already has the information available beforehand, it is a *query/response* transaction. If the responder does not have the information, but no pre-editor context validation is required before processing, the transaction is a *request/confirm* one. If the latter is required, the next question is whether the transaction results in a residual obligation to fulfill terms of a contract. If so, it is a *commercial transaction*. Otherwise it is a *request/response* transaction. These types of business transactions cover all known legally binding interactions between two decision making applications as defined in Open-edi (ISO, 1995).

Owing to the strict definition, a UMM *business transaction* follows always the same pattern: A *business transaction* is performed between two *authorized roles* that are already known from the *business transaction use case* and that are assigned to exactly one *business transaction swimlane* each. Each *authorized role* performs exactly one activity. The object flow between the *requesting* and the *responding business activity* is mandatory. The object flow in the reverse direction is optional. In our example the *business transactions* `request for quote` (6) and `place order` (7) both are two-way transactions. `Request for quote` is of type *request/response* and `place order` is a *commercial transaction*.

The requirements described in a *business collaboration use case* are choreographed in the activity graph of a *business collaboration protocol*, which is defined in a *business choreography view*. This one-to-one relationship is denoted by another *maps to* dependency. In our example, the `order from quote` requirements (1) are mapped to the homonymous *business collaboration protocol* (5). A *business collaboration protocol* choreographs a set of *business transaction activities* and/or *business collaboration activities*. A *business transaction activity* is refined by the activity graph of a *business transaction*. In our example, the *business collaboration protocol* of `order from quote` (5) consists of two *business transaction activities*: `request for quote` and `place order`. Each of them is refined by its own *business transaction* (6 and 7). *Maps to* dependencies keep track of this refinement.

The *business information views* are used to define the structure of *business documents* exchanged in *business transactions*. Each of the four *information envelopes* exchanged in the two *business transactions* leads to a *business information view* describing the envelopes document structure. Since modeling the *business documents* is out of scope for this paper we do not detail them any further.

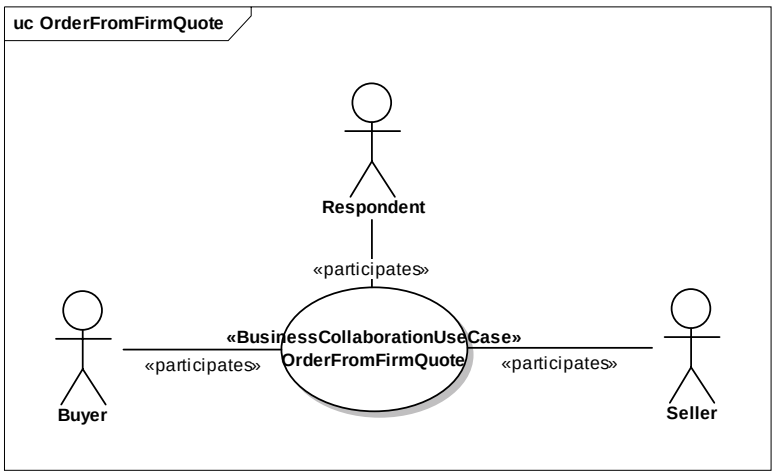

Fig. 2 Multi-pary Business Collaboration Use Case: Order from Firm Quote

4 Problem: Nested Business Transactions

In the example of section 3, the *business collaboration use case* `order from quote` involves exactly two *authorized roles*. In this case we speak of a bi-lateral collaboration. Next we are going to extend our example to a multi-party collaboration. We assume that the wholesaler will contact the retailers bank to check his credit before making a firm quote. This means we have three *business partner types*: `retailer`, `wholesaler` and `bank`. These three *business partner types* collaborate in a *business collaboration use case* `order from firm quote` (see figure 2): the `retailer` acts as `buyer`, the `wholesaler` acts as `seller`, and the `bank` acts as `respondent`.

Evidently, we need another *business transaction* that is realized between the wholesaler and the bank to check the retailers credit. We already learned for the sake of reuse that a *business transaction* is not defined between *business partner types*, but between more abstract *authorized roles*. The left hand side of figure 3 depicts the *business transaction use case* `check credit` that involves a `check requestor` and a `respondent`. The right hand side shows the corresponding *business transaction* between these *authorized roles*. The artifacts of figure 3 and their model elements must be added to the UMM model of figure 1.

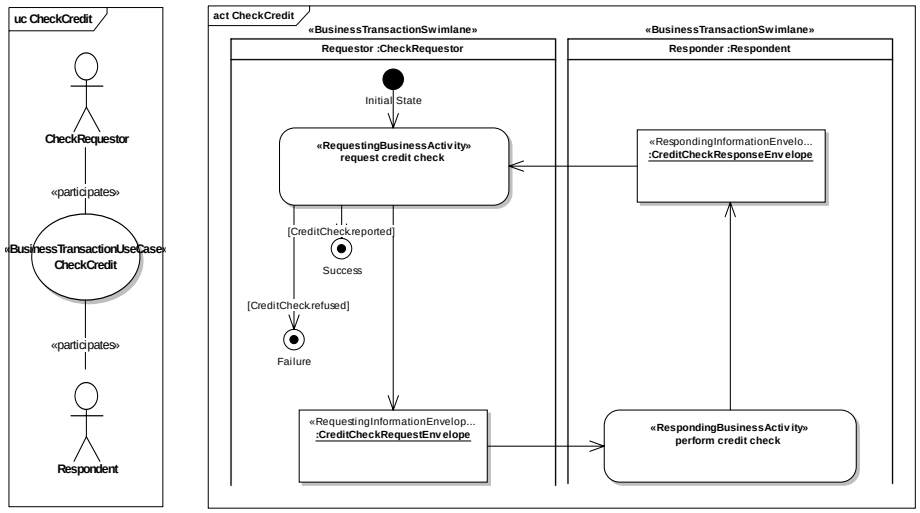

Fig. 3 Business Transaction Use Case and Business Transaction: Check Credit

Naively, one may think that the *business transaction use case* `check credit` is directly included in the new *business collaboration use case* `order from firm quote`. In this case, a *business transaction activity* `check credit` would have to be added to the corresponding *business collaboration protocol* `order from firm quote`. It follows that we would have to specify a choreography among the three *business transaction activities* `request for quote`, `place order`, and `check credit`. However, this is an impossible task.

From a business point of view the scenario is as follows: First the buyer submits his quote. In order to decide whether to make a firm quote or not, the seller requests the bank to check the buyers credit. After receiving the result of the credit check from the

bank, the seller returns a quote document, which either includes the quote or a reason for quote rejection.

In the *business collaboration protocol* we have to define the control flow between the `request for quote` and the `check credit` *business transaction activities*. In UML, the transition from one activity to another one is triggered by the fact that the first one is completed. Looking at our example, it is clear that the `request for quote` *business transaction activity* is started first. However, it is not completed before the `check credit` *business transaction activity* starts. In fact, `check credit` is nested within `request for quote`. This means that the nested activity is triggered as part of starting the encompassing activity and the nested one has to complete for the encompassing one to finish. In case of nested *business transaction activities*, it is not possible to specify a transition between the *business transaction activities* in the *business collaboration protocol*. In figure 4 we depict our example case where the transition between `request quote` and `order from quote` cannot be defined.

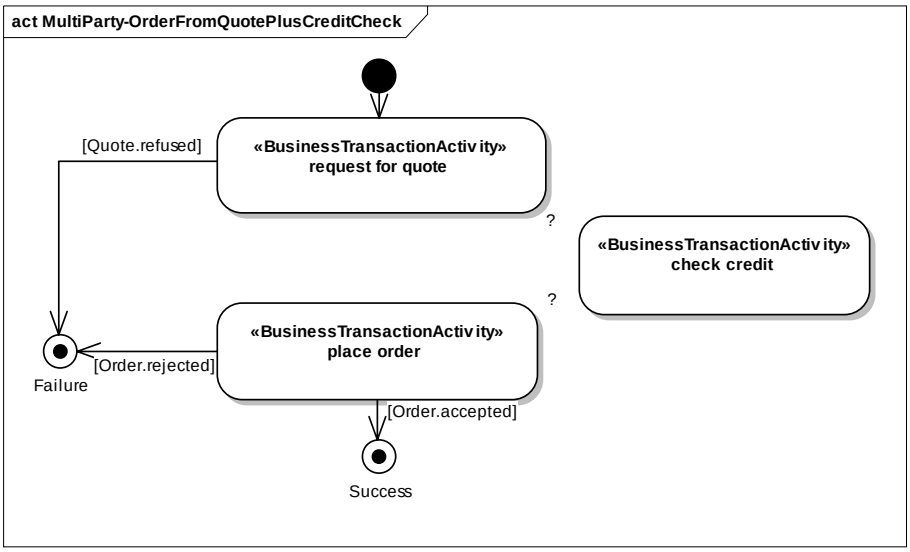

Fig. 4 Undefined Transaction in Business Collaboration Protocoll: Order from Firm Quote

Nested *business transactions* are not necessary when modelling bi-lateral collaborations. According to the *business transaction* semantics - which are derived from the Open-edi reference model (ISO, 1995) - a responder has first to reply in a two-way *business transaction* before continuing with another business transaction in the same business collaboration. Therefore, nested *business transactions* usually appear in multi-party collaborations. These nested transactions are very typical in multi-party supply chains. Since *business collaboration protocols* are only able to handle multi-party collaborations without nested transactions, they are not appropriate for real-world multi-party scenarios. Thus, the current UMM is more or less limited to bi-lateral collaborations.

In order to overcome the limitations of nested transactions, one may think to split the encompassing two-way *business transaction* into two one-way *business transac-*

tions: One that starts before the nested one and one that completes after the nested one is finished. However, this would violate the rules of the *business transaction* semantics as defined in Open-edi (ISO, 1995). The four two-way business transaction types *request/response*, *query/response*, *request/confirm* and *commercial transaction* have a certain business semantic and cannot be split into two *business transactions* of type *notification* or even *information distribution*.

It follows that a multi-party business collaboration must be split into bi-lateral business collaborations. As shown in figure 5, the *business collaboration use case* `order from firm quote` is built by two included *business collaboration use cases*: `order from quote` (which we detailed in Section 3) and `check creditworthiness`. The *business collaboration use case* `check creditworthiness` is simple and includes only one *business transaction use case*: `check credit`. Due to space limitations and the triviality we do not depict this include association in figure 5.

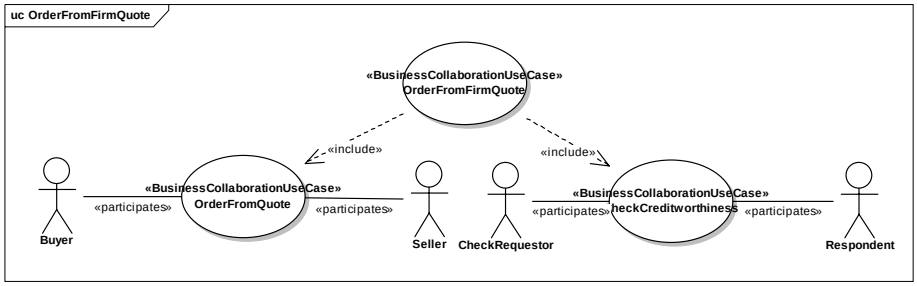

Fig. 5 Splitting a Multi-party Collaboration into Bi-lateral Collaborations

5 A UML Profile for Local multi-party Choreographies

A limitation of splitting up a multi-party business collaboration into multiple bi-lateral business collaborations is the fact, that all dependencies between the different bi-lateral business collaborations are lost. In case of our example, this means that the dependencies between `request for quote` and `check creditworthiness` are absent in the resulting UMM model. Accordingly, the information that checking the buyers credit is nested within the request for quote is lost. From a UN/CEFACT perspective this seems to be ok, since UN/CEFACTs goal is providing standardized reference models for the collaborative space. Evidently, it is not intended to standardize how enterprises and organizations work internally. Accordingly, mandating a credit check as part of a request for quote is not the task of UN/CEFACT. It is an internal decision made by a seller.

Even if it is not UN/CEFACTs task to model the internal decisions of a party, we think it is critical to provide the party with guidelines on how to model its local choreography in alignment with the global choreography provided in UMM models. These guidelines may be used by an individual party, but may also be used by a supply chain designer who is able to suggest a certain behavior to the supply chain partners. For this purpose we have developed a UML profile for local choreographies extending UMM.

The UMM is already prepared for such extensions since the meta model is divided into a set of meta modules. The basic UMM concepts are defined in the base module and in the foundation module (Dietrich *et al.*, 2006). Both, extension modules and specialization modules provide add-on concepts to the foundation. The specialization modules are developed and maintained by UN/CEFACT, whereas extension modules are created and maintained by external organizations. Thus, the profile described in this paper is an example of an extension module. However, we consider submitting the proposal to UN/CEFACT in order to progress it to a specialization module.

5.1 Steps of specifying a local choreography by example

In order to explain the concepts of our UML profile for local choreographies, we first have a look at our example before looking at the stereotype definitions and the meta model constraints. We demonstrate the local choreography by means of the `seller` in our `purchase management` example. Since this local choreography does not fit in any of the three main UMM views, it is placed in another top level package that is stereotyped as *local choreography view*. It is important to note that a *local choreography view* must always be defined in the same model as the relevant business collaborations and business transactions of the global choreography. This does not necessarily mean that the artifacts of the global choreography have originally been developed as part of this model. The global choreography or parts thereof may be provided in a registry and imported into a model that only adds the *local choreography view*.

Within the *local choreography view* we define an activity that represents the overall choreography of the `seller` in `order from firm quote`. This activity is stereotyped as *local choreography*. The flow within this *local choreography* is depicted in the activity diagram of figure 6. The main building blocks are those *requesting/responding business activities* that are carried out by the `seller`. A detailed look on the *business transactions* in figure 1(6,7) shows that the *requesting/responding business activities* are assigned to the partitions of the *authorized roles* in the *business transactions* (`quote responder`, `order responder`), but not to the *authorized roles* in the collaboration (`seller`). However, there exist two *maps to* dependencies each starting from the `seller` in the *collaboration requirements view* (1): one to `quote responder` in one *transaction requirements view* (2) and one to `order responder` in the other *transaction requirements view* (3). Hence, we are able to calculate the *requesting/responding business activities* of the `seller`. In the `order from quote` collaboration with the `buyer`, the `seller` carries out two *responding business activities*: `calculate quote` and `act on purchase order`.

These two *responding business activities* are part of the local choreography of the `seller`. The next step is to specify the flow among them. We are able to derive the flow from the *business collaboration protocol* of `order from quote` (see figure 1, (5)). The *business collaboration protocol* includes only *business transaction activities* (and *business collaboration activities*). These activities are executed by exactly two *authorized roles*. In a bi-lateral business collaboration each *authorized role* must be involved in each of the activities. It follows that the flow in the *local choreography* is similar to the one in the *business collaboration protocol*. However, each *business transaction activity* is replaced by the *requesting/responding business activity* of the respective *authorized role* in the underlying *business transaction*. The *business collaboration protocol* in figure 1 (5) shows a sequence of `request for quote` and `place order`. Accordingly, the *local choreography* of the `seller` in figure 3 is built by the sequence of `calculate quote`

(sellers task in `request for quote`) and `act on purchase order` (his task in `place order`). The transitions and their guard conditions are the same for the *business collaboration protocol* and the *local choreography*. This means that a `refused quote` (note we write short `Quote.refused` instead of `Quote.oclInState(refused)`) leads to a failure state. The transition between `calculate quote` and `act on purchase order` is guarded by the fact that the `quote` was `provided`. An `rejected order` leads to a failure state after `act on purchase order`, whereas an `accepted order` leads to a success.

The next step is detailing each of the *requesting/responding business activities* in the *local choreography*. An object node is added for each incoming and outgoing *information envelope*. *Requesting/responding business activities* taken from two-way transactions have both an input and an output node. If they are part of a one-way *business transaction*, a *requesting business activity* has only an output node and a *responding business activity* has only an input node. In our example, `calculate quote` and `act on purchase order` are both taken from a two-way *business transaction*. Thus, they have an input and an output node. The *information envelopes* assigned to these nodes correspond to the input and output defined in the *business transactions* in figure 1 (6,7).

For each input node we add an *accept event action* that is stereotyped as *receive business information*. It is used to recognize the event of an incoming *information envelope* and to hand it over to the *requesting/responding business activity*. In case of a *responding business activity* this event and the resulting transfer of the *information envelope* is required to start the *responding business activity*. In our example of figure 6 the overall *initial state* leads immediately to the `calculate quote` activity. However, before the first activity within `calculate quote` is started, the *accept event action* `receive quote request` must recognize the receipt of a `quote request envelope` and transfer it to `calculate quote`.

The flow within a *requesting/responding business activity* is modeled by the following concepts: *Private actions* and *private activities* are used to model tasks that are internal to the organization and are not visible to other parties. *Private activities* may be decomposed into further activities and actions, *private actions* do not. The next concept is *send business information*. It is a special kind of the *send signal action* that is used to send an *information envelope* to the other *authorized role*. However, the other *authorized role* is not part of a *local choreography* - this exchange is already defined in the corresponding *business transaction*. In the *local choreography*, *send business information* simply writes the *business information envelope* to the output node of the *requesting/responding business activity*. In our example, the *send business information* `send quote response` transfers the `quote envelope` to the output node of `calculate quote`. The fact, that the *quote envelope* is returned to *obtain quote* (executed by the *quote requestor*) is not shown in the *local choreography* of figure 6 - it is already defined in the `request for quote` *business transaction* of figure 1 (6).

So far, we have not shown how to overcome the problem of nested transactions. For this purpose we use the concept of a *call nested transaction*. As we know from our example, checking the customer credit has to be done after receiving a quote request and before responding to it. This means that the *business transaction* `check credit` is started as part of the sellers `calculate quote` activity. Accordingly, we define the concept of a *call nested transaction* as a special kind of the UML *call action behavior*, used to call another structured activity - which is a *requesting/responding activity* of the same *authorized role*. In our example of figure 6, the `calculate quote` activity includes the *call nested transaction* `request credit check`. This one calls the synonymously named *requesting business activity* `request credit check` - which is the `sellers`

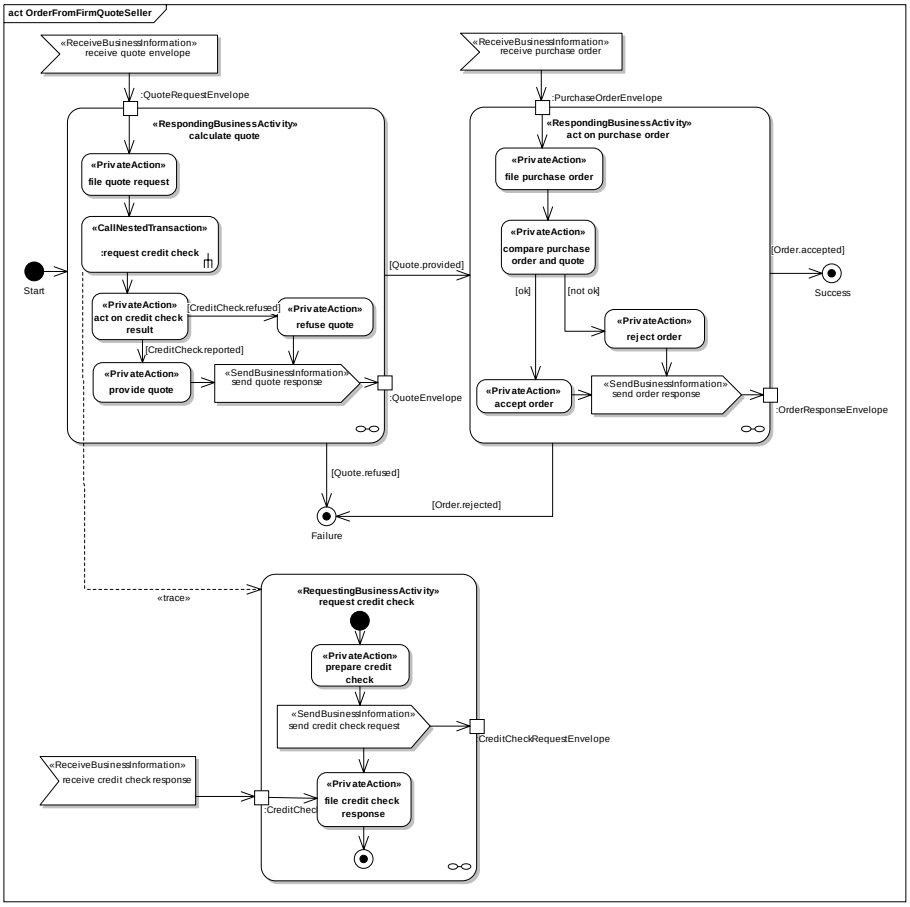

Fig. 6 Local Choreography: Seller in Order from Firm Quote

task in the *business transaction* `check credit` (see figure 3). We again specify a flow within the *requesting business activity* `request credit check`. After finishing this flow, control is given back to `calculate quote`. Since `request credit check` is a *requesting business activity*, its internal flow first includes a *send business document* action before receiving a return back. However, its flow is only able to continue with `file credit check`, if a `credit check response envelope` is recognized by the *receive business document action* `receive credit check response`. Furthermore, it should be noticed that *call nested transaction* is only used for calling a single *requesting/responding business activity*. If a whole collaboration is nested, the *call nested collaboration* concept is used. This one calls another *local choreography*.

5.2 Stereotypes and Constraints of the profile for local choreographies

In the previous subsection we have demonstrated the methodology of our UML profile for local choreographies by means of our simple example.In this example we already

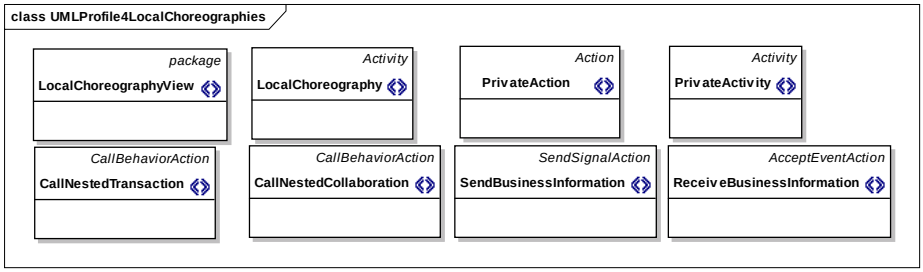

Fig. 7 Stereotypes of our UML Profile for Local Choreographies

made references to the following stereotypes of our UML profile for modeling local choreographies (see figure 7):

The stereotype *local choreography view* is a new type of a top level package in a UMM business collaboration model. It captures all the artifacts necessary to specify local choreographies. The stereotype *local choreography* extends the concept of an activity. This activity is carried out by one organization only. The activity is decomposed into a controlled flow of *requesting/responding business activities* performed by the corresponding organization. Further stereotypes are defined for modeling the flow of a *requesting/responding business activity*. The stereotypes *private action* and *private activity* are used to model actions and activities of this flow that are not visible to external organizations. Elements of these two stereotypes are not necessarily atomic - a *private activity* is decomposed into other activities/actions and a *private action* may call a sub-process.

The stereotype *call nested transaction* is another kind of action in this flow. It is used to call a flow in another *requesting/responding business activity* of the corresponding organization. The stereotype *send business information* extends the concept of a UML *send signal action* and is also part of this flow. It is used to denote sending an *information envelope* to another organization. This *information envelope* is assigned to the output node of the corresponding *requesting/responding business activity*. The stereotype *receive business information* is a special kind of a UML *accept event action*. It is used to recognize an incoming *information envelope*. It is part of a *local choreography*, but does not sit in the flow of a *requesting/responding business activity*. However, the received *information envelope* is given to an input node of the *requesting/responding business activity*. Note, the stereotypes *requesting business activity*, *responding business activity*, and *information envelope* are already defined in UMM.

Furthermore, the stereotypes have certain relationships with each other. These relationships are specified by means of constraints on the meta model. These constraints are defined by the means of the object constraint language (OCL). Due to page limitations we only include their plain text descriptions:

- Since a *local choreography view* is a top level package, the parent of a *local choreography view* is a *business collaboration model*.
- A *local choreography* is directly specified within a *local choreography view*. Consequently, the parent of a *local choreography* must be a *local choreography view*. A *local choreography view* may include multiple *local choreographies* as its children.
- A *local choreography* specifies a flow amongst *requesting/responding business activities* and includes *receive business information actions*. The children of a *local*

choreography must be either *requesting/responding business activities* or *receive business information actions* - no other children are allowed.
- All *requesting/responding business activities* within a *local choreography* are performed by the same organization. Consequently, the activities within a *local choreography* must all be in the *business transaction swimlanes* of *authorized roles* in *business transactions*, that all have a (recursive) mapping to the same *authorized role* in a *business collaboration use case*.
- The transitions between *requesting/responding business activities* in a *local choreography* must correspond - including their guards - to the transitions in the *business collaboration protocol* from which the *requesting/responding business activities* are taken. Note, in fact they are taken from the *business transactions* refining the *business transaction activities* of the *business collaboration protocol*.
- A *requesting business activity* coming from a one-way *business transaction* has only one output node, one from a two-way *business transaction* has an output and an input node. A *responding business activity* coming from a one-way *business transaction* has only an input node, one from a two-way *business transaction* has an input and an output node. The *information envelopes* of the input and output nodes must be consistent to the object flows specified in *business transactions*.
- Output nodes of a *requesting/responding business activity* do not show any further object flow in the *local choreography*. This flow must be specified in a *business transaction*.
- A *receive business document* action must lead to an input node of a *requesting/responding business activity*. A *send business document* action must lead to an output node of a *requesting/responding business activity*.
- A *requesting/responding business activity* includes children of the following stereotypes: *private action*, *private activity*, *send business document*, *call nested transaction*, and *call nested collaboration* - no others are allowed. Vice versa, the parent of these stereotypes must always be a *requesting/responding business activity*.
- A *call nested transaction* must always call a *requesting/responding business activity*. A *call nested collaboration* must always call another *local choreography*. No recursive circles are allowed.

6 Conclusions

In this paper we build up-on the UN/CEFACT modeling methodology (UMM). UMM is defined as a UML profile for modeling global choreographies. A UMM business collaboration model captures the commitments and agreements on the flow of business information between two parties. We used a simple order from quote example to demonstrate modeling a bi-lateral, global choreography by means of UMM. More complex and realistic UMM choreographies are part of all *business requirements specifications* developed for international trade by UN/CEFACT , as well as by national e-government frameworks, such as XV in Germany, GovDex in Australia, and the Governments of Canada Strategic Reference Model (GSRM).

It is the intention of UN/CEFACT to use UMM for specifying reference models that partners can agree up-on. It is not the intention of UN/CEFACT to standardize the business processes within an organization. Accordingly, UMM is not capable of modeling such processes. If a partner in one bi-lateral collaboration decides to contact another partner in another bi-lateral collaboration, this is considered as an internal

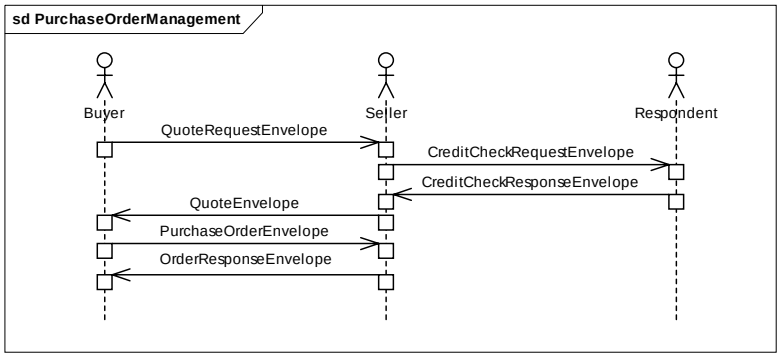

Fig. 8 Multi-party Message Exchanges: Order from Firm Quote

decision in the internal business process of that partner. Thus, UMM has its limitations in modeling multi-party collaborations. Again, we used our simple demonstration example of order from quote between a seller and a buyer to highlight the problems of a nested transaction between the seller and a bank.

However, it is critical for an organization to model its own business processes and at the same time being in-line with business collaboration models the organization agreed up-on. Similarly, a supply chain designer, who is able to recommend certain tasks to the supply chain partners, may find it critical to model multi-party collaborations. These were the drivers for extending the UMM for the purpose of modeling local choreographies as well as multi-party collaborations. Nevertheless, our proposed UMM extension may be used no matter whether two or more business partners collaborate and no matter whether nested transactions are involved or not. Our approach extends the UMM by another set of stereotypes. In this profile, nested collaborations are realized by calls to subprocesses representing these nested collaborations. However, a total of eight stereotypes as well as a well defined set of constraints between them and to stereotypes of the UMM ensure a consistent modeling approach from global to local choreographies. The resulting profile is also considered as a domain-specific customization of UML that simplifies code development in a next step. Furthermore, we demonstrate the methodological steps of our approach by means of an example.

In order to support our approach, we created the UML profile definition for the commercial UML tool Enterprise Architect. Accordingly, all stereotypes are then built into Enterprise Architect and the profile may be used together with our Enterprise Architect Add-in for the UMM foundation module (Hofreiter *et al.*, 2007 b). However, full fledged user support that goes beyond regular modeling features - like in the previously mentioned Add-in - is open to future work. Nevertheless, it should be noted that out approach sits on top of the UML meta model and, thus, any other UML tool may be used to create compliant models. This fact helped also in the early stages of developing the UML profile: The Australian GovDex project reported the need for nested transactions. We offered a preliminary version of our profile for local choreographies in order to complement the UMM approach already in use by GovDex. Even if no specific tool support was available at this time, the GovDex project was able to apply the plain Enterprise Architect features to test our proposal. The feedback loops with GovDex helped a lot in the evaluation of the proposed UML profile for local choreographies which finally proved to provide a useful extension to UMM.

The approach proposed in this paper shows how to model a local choreography of a partner. The constraints specified in our profile guarantee that the local choreography respects the commitments made in the global choreography - assuming that the global choreography is correct. Having defined a local choreography it is also possible to automatically derive a sequence of message exchanges for a multi-party collaboration. The derivation process starts off with a bi-lateral collaboration. If one partner calls a nested transaction or a nested collaboration in its local choreography, the additional partnership is added to the message flow. The implementation of this automatic derivation is up to future work. Figure 8 shows the message exchanges in a successful case of our `order from firm quote` example.

References

Andrews Tony, Curbera Francisco, Dholakia Hitesh, Goland Yaron, Klein Johannes, Leymann Frank, Liu Kevin, Roller Dieter, Smith Doug, Thatte Satish, Trickovic Ivana, Weerawarana Sanjiva (2003) Business process execution language for web services. Version 1.1, http://www-128.ibm.com/developerworks/library/specification/ws-bpel/

Barros Alistair P., Dumas Marlon, Oaks Phillipa (2005) Standards for web service choreography and orchestration: Status and perspectives. In: Proceedings of the Business Process Management (BPM) Workshops. pp. 61–74

Booch Grady, Rumbaugh James, Jacobson Ivar (2005) The Unified Modeling Language User Guide. Addison Wesley. 2nd Edition

Decker Gero, Kopp Oliver, Leymann Frank, Weske Mathias (2007) Bpel4chor: Extending bpel for modeling choreographies. In: Proceedings of the 2007 IEEE International Conference on Web Services (ICWS 2007). IEEE Computer Society. pp. 296–303

Dietrich Jens, Hofreiter Birgit, Huemer Christian, Liegl Philipp, Schuster Rainer, Zapletal Marco (2006) Un/cefact's modeling methodology (umm), umm meta model - foundation module. Technical Specification V1.0, http://www.unece.org/cefact/umm/UMM_Foundation_Module.pdf

Hofreiter Birgit, Huemer Christian, Kim Ja-Hee (2006 a) Choreography of ebxml business collaborations.. Inf. Syst. E-Business Management 4(3): 221–243

Hofreiter Birgit, Huemer Christian, Liegl Philipp, Schuster Rainer, Zapletal Marco (2007 a) Deriving executable bpel from umm business transactions. In: Proceedings of the 2007 IEEE International Conference on Services Computing (SCC 2007). IEEE Computer Society. pp. 178–186

Hofreiter Birgit, Huemer Christian, Liegl P., Schuster R., Zapletal Marco (2006 b) Un/cefact's modeling methodology (umm): A uml profile for b2b e-commerce.. In: Advances in Conceptual Modeling - Theory and Practice, ER 2006 Workshops. Vol. 4231. Springer LNCS

Hofreiter Birgit, Huemer Christian, Liegl Philipp, Schuster Rainer, Zapletal Marco (2007 b) Umm add-in: A uml extension for un/cefact's modeling methodology. In: Service-Oriented Computing - ICSOC 2007, Fifth International Conference. Vol. 4749. Springer LNCS. pp. 618–619

Hofreiter Birgit, Huemer Christian, Zapletal Marco (2006 c) Registering umm business collaboration models in an ebxml registry. In: Proceedings of the 8th IEEE International Conference on E-Commerce Technology (CEC 2006) / 3rd IEEE International Conference on Enterprise Computing (EEE 2006). IEEE Computer Society

ISO (1995) Open-edi reference model. ISO/IEC JTC 1/SC30 ISO Standard 14662

Kavantzas Nickolas, Burdett David, Ritzinger Gregory, Fletcher Tony, Lafon Yves, Barreto Charlton (2005) Web services choreography description language. Version 1.0, http://www.w3.org/TR/ws-cdl-10/

Khalaf Rania (2007) From rosettanet pips to bpel processes: A three level approach for business protocols. Data and Knowledge Engineering

Korherr Birgit, List Beate (2006) Extending the uml 2 activity diagram with business process goals and performance measures and the mapping to bpel. In: Advances in Conceptual Modeling - Theory and Practice, ER 2006 Workshops. Vol. 4231. Springer LNCS. pp. 7–18

Kraemer Dave, Yendluri Prasad (2002) Realizing the benefits of implementing rosettanet implementation framework (rnif) version 2.0. http://www.rosettanet.org/cms/export/sites/default/RosettaNet/Downloads/whitePapers/RNIF2finalv3.pdf

Kramler Gerhard, Kapsammer Elisabeth, Kappel Gerti, Retschitzegger Werner (2005) Towards using uml 2 for modelling web service collaboration protocols. In: Proceedings of the First International Conference on Interoperability of Enterprise Software and Applications (INTEROP-ESA'05)

Lee Ronald M. (2000) Documentary petri nets: A modeling representation for electronic trade procedures. In: Business Process Management: Models, Techniques, and Empirical Studies. Springer

Lenz Kirsten, Oberweis Andreas (2003) Inter-organizational business process management with xml nets. In: Petri Net Technology for Communication-Based Systems. Vol. 2472. Springer LNCS. pp. 243–263

Leymann F., Roller D., Schmidt M.-T. (2002) Web services and business process management. IBM Systems Journal - New Developments in Web Services and E-commerce

Ling Sea, Loke Seng Wai (2002) Advanced petri nets for modelling mobile agent enabled interorganizational workflows. In: Proceedings of the 9th IEEE International Conference on Engineering of Computer-Based Systems (ECBS 2002). IEEE Computer Society. pp. 245–252

List Beate, Korherr Birgit (2005) A uml 2 profile for business process modelling. In: ER 2005 Workshops Proceedings

Peltz Chris (2003) Web services orchestration and choreography. IEEE Computer 36(10): 46–52

Penker Magnus, Eriksson Hans-Erik (2000) Business Modeling With UML: Business Patterns at Work. Wiley

Piccinelli Giacomo, Emmerich Wolfgang, Zirpins Christian, Schütt Kevin (2002) Web service interfaces for inter-organisational business processes: An infrastructure for automated reconciliation. In: Proceedings of the 6th International Enterprise Distributed Object Computing Conference (EDOC 2002). IEEE Computer Society. pp. 285–292

Schatz Willie (1988) Edi: Putting the muscle in commerce and industry. Datamation

Stitzer Alan, Crawford Mark (2003) Core components technical specification - part 8 of the ebxml framework. Version 2.01, http://www.unece.org/cefact/ebxml/CCTS_V2-01_Final.pdf

UN/CEFACT (2003) Un/cefact - ebxml business process specification schema. Version 1.10, http://www.untmg.org/dmdocuments/BPSS_v110_2003_10_18.pdf

van der Aalst Wil M. P. (2002) Inheritance of interorganizational workflows to enable business-to-business. Electronic Commerce Research 2(3): 195–231

van der Aalst Wil M. P., Weske Mathias (2001) The p2p approach to interorganizational workflows. In: Advanced Information Systems Engineering, 13th International Conference, CAiSE 2001. Vol. 2068. Springer LNCS. pp. 140–156

van der Aalst Wil M.P. (1999) Interorganizational workflows: An approach based on message sequence charts and petri nets. Systems Analysis - Modelling - Simulation 34(3): 335–367

White Stephen A. (2006) Business process modeling notation specification 1.0. http://www.omg.org/docs/dtc/06-02-01.pdf