Faculty of Computer Science

universität wien

Department of Distributed and Multimedia Systems

Technical Report TR-20080205 February 2008

A UML Profile For Nested Business Collaborations

Birgit Hofreiter,

A UML Profile for Nested Business Collaborations

Birgit Hofreiter

Department of Information Systems, University of Technology Sydney and
Department of Distributed and Multimedia Systems, University of Vienna
birgit.hofreiter@univie.ac.at

Abstract. Today, most approaches for inter-organizational business processes start bottom-up from the interfaces and the workflows of each partner described on the IT layer. Alternatively, one may start from the commitments and agreements between business partners to reach their complementary business goals. The latter approach is target of the UN/CEFACT Modeling Methodology (UMM). Due to its focus on agreements and commitments that are always made on a bi-lateral basis, UMM is limited to business processes between two partners only. However, in a model driven approach the UMM artifacts must be further elaborated towards an IT solution. A next step would be to model the commitments of a partner in the middle of a supply chain, i.e. to describe the local choreography of a partner that consolidates the different bi-lateral models into a multi-party choreography. Since UMM does not support this step, we develop a UML profile for nested business collaborations. Furthermore, we define relationships of the stereotypes of this UML profile with the existing UMM stereotypes.

Keywords: Business Process Modeling, Choreography, UML, Inter-organizational BP

1 Motivation

Inter-organizational systems have been implemented for many years by the concepts of electronic data interchange (EDI) [32]. Whereas maintaining EDI partnerships is considered complex and expensive, the trend towards Web Services as technology of choice for distributed systems gave new hopes to the community. In contrary to the pure data centric EDI standards, Web Services specifications also care about the business processes.

In Web Services discussions two terms popped up: orchestration and choreography [5, 28]. Both are closely related, but must be well distinguished to follow this paper. 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 partner's point of view. It makes the public parts of its local process visible to others. A global choreography defines the inter-organizational process from a neutral perspective. A global choreography has the potential to achieve an agreement between the partners. Local choreographies enable the configuration of each partner's system.

In Web Services most attention is spent on executable orchestrations and less attention on local orchestrations. The business process execution language (BPEL) is able to handle both. Global choreographies - as described by the choreography definition language (WS-CDL) - play a rather secondary role. This is also due to the IT-centric focus of Web Services. The workflow of internal business processes is described by its orchestration. This workflow is also the starting point for the definition of inter-organizational processes. The local choreography of a business partner may be calculated by a projection on the orchestration considering only those activities that are visible to the outside world. Inasmuch business collaborations are built bottom-up.

If business partners want to collaborate they must have complementary local choreographies - which is rather unlikely if the choreographies have been developed in isolation. This means a big company announces its local choreography and all its smaller business partners adopt their interfaces accordingly. If this is not the case and business partners have non-complementary processes, they have to undergo the cumbersome procedure of harmonizing their local choreographies and adjusting their orchestrations that are bound to the local choreographies.

A top-down approach to business collaborations provides an alternative to the previously described scenario. An analysis may start with the economic drivers for the business collaboration. This means describing the business models by means of the economic values that are exchanged between the business partners. Approaches to be used on this level of abstraction are e3-Value [9], REA [8] or BMO [27].

In order to guarantee that each partner deserves its economic value they have to agree with each other on the inter-organizational business processes to realize the value exchanges. The resulting global choreography becomes a kind of contract guiding the business partnership. An approach on this level of abstraction is delivered by the United Nations Centre of Trade Facilitation and e-Business and their UN/CEFACT Modeling Methodology (UMM). The UMM specification is defined as a UML profile [14], which we have co-edited. Usually, commitments are made on a bi-lateral basis. Accordingly, UMM models always describe business collaborations between two parties. Also similar to a contract, a UMM model describes the commitments and agreements from a neutral perspective. In summary, the UMM provides a methodology for bi-lateral and global choreographies.

In order to realize a business value a business partner must most likely interact in a certain business case with a lot of different business partners. Each of the bi-lateral collaborations may be based on a UMM reference model. However, it is the local decision of the business partner under consideration to fix with which other parties to interact and how to nest the different bi-lateral collaborations. Consequently, a next step in the top-down approach is the definition of a local and multi-party choreography for a certain business partner that respect the commitments made on the level of global choreographies.

Further steps are the definition of the orchestration of the internal business processes that are bound to the local choreographies as well as the automatic generation of machine-interpretable workflow descriptions of these orchestrated processes.

In order to allow a straight-through modeling approach, it is desirable to base the top-down approach on a single modeling paradigm. UMM - which we selected as core standard for our approach - is already defined as a UML profile. On the preceding business model level it should be mentioned that REA is about to be integrated into UMM and a UML profile for e3-Value has recently been proposed [16]. In order to complement the overall vision we propose a UML profile for modeling the local multi-party choreography that is in alignment with the global choreography of UMM. The dedicated binding to the upper layers serves as a justification for developing yet another choreography language. We already integrate some concepts for modeling orchestrations. However, the current focus of or profile is on local choreographies, the suitability for orchestrations has to undergo further testing.

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. Next we demonstrate UMM's limitations to model multi-party choreographies by extending this example. Section 4 uses this example to demonstrate how our proposed UML profile for local choreographies overcomes this problem. In Section 5 we present the underlying meta models - first of the relevant UMM parts followed by a detailed discussion of the meta model for local choreographies. Section 6 summarizes the paper.

2 Related Work

The idea of defining business processes crossing organizational boundaries goes back to ISO's Open-edi reference model [17]. A first implementation of the choreography aspects of this model was a Petri-Net approach contributed by Lee [22]. Also other authors used Petri-Nets to define the workflow between organizations [23,25,1]. 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) [6] and the Business Process Modeling Notation (BPMN) [35]. The main advantage of both 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 [20,26,29]. Beside UMM, UML customizations for modeling inter-organizational processes are described in the RosettaNet Framework [31] and in Kramler et al. [21]. 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) [4,24] 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 [19] and by ourselves for UMM [12]. In the area of Web Services, the Web Services Choreography Description Language (WS-CDL) [18] 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 [7] has recently been proposed to extend BPEL for describing global choreographies. Another XML-based language for describing global choreographies is ebXML's Business Process Specification Schema (BPSS) [33]. Since the BPSS is based on UMM and is more or less an XML representation of UMM's business transaction view [10], 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 work-around called *complex business transaction activity* that works in simple scenarios.

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 [2]. 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 [1,3]. Piccinelli et al.[30] propose another transformation approach in particular for peer-to-peer collaborations.

3 UN/CEFACT's Modeling Methodology (UMM)

The UN/CEFACT is a standards organization that became known by creating and maintaining the UN/EDIFACT standard. Since UN/CEFACT's 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 [34]. UN/CEFACT's Modeling Methodology (UMM) models the choreography and data exchange commitments independent of the IT.

The UMM methodology consist of three main views for modeling business collaborations. Firstly, the Business Domain View (BDV) provides a framework for understanding existing business processes and categorizing these business processes into business areas and process areas. Secondly, the Business Requirements View (BRV) identifies possible business collaborations and further elaborates on these collaborations. It describes processes and resources used to achieve certain objectives, and the resulting commitments. Thirdly, the Business Transaction View (BTV) defines the business documents being exchanged and the order of these exchanges, i.e. the choreography of the business collaboration.

Each of these views comprises a number of different well defined UML artifacts. These artifacts must follow the UMM meta model that is defined as a UML profile. Accordingly, the UMM meta model specification covers a set of well defined stereotypes including tagged definitions for each of the above mentioned views.

Due to page limitations we are not able to elaborate in detail on all features of the UMM. We limit ourself to those concepts necessary to understand our approach for nested collaborations and local choreographies. These means we concentrate only on those artifacts and stereotypes that have a dependency with the proposed local choreography. Thus, we introduce only a small part of the BRV, but most of the artifacts of the BTV. The reader interested in more UMM details is referred to the technical specification [14] which we have co-edited and to our publication in [11] detailing many UMM backgrounds.

3.1 UMM by example

For a better understanding we first demonstrate the relevant UMM artifacts by example in this section. Later on in section 5 we detail the underlying meta model. The demonstrating example is an order from quote business collaboration between a buyer and a seller. This example is designed to be as simple as possible for an easy understanding and as complex as necessary to understand the proposed approach. It should be noted that we even hide some UMM specifics that are irrelevant for this paper, but are well described in [11].

Early steps in the BDV and first steps in the BRV have shown a possible collaboration between a buyer and a seller in an order from quote, which consists of two subprocesses: obtaining a quote and placing an order. This fact is modeled in the *collaboration requirements view* - a subview of the BRV (see Fig. 1). The requirements of the collaboration between the *authorized roles* `buyer` and `seller` are described in the *business collaboration use case* `order from quote`. A *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` includes the *business transaction use cases* `request for quote` and `place order`.

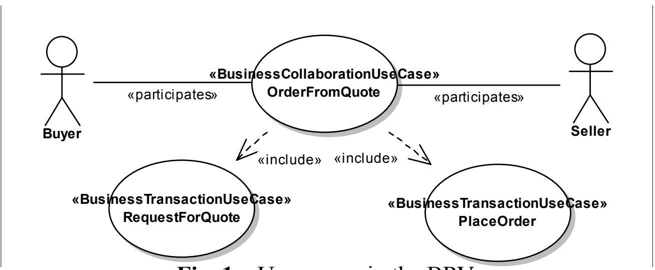


Fig. 1. Uses cases in the BRV

The BTV - the third view - builds upon these use cases in order to define 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* - which is part of the *business interaction view*.

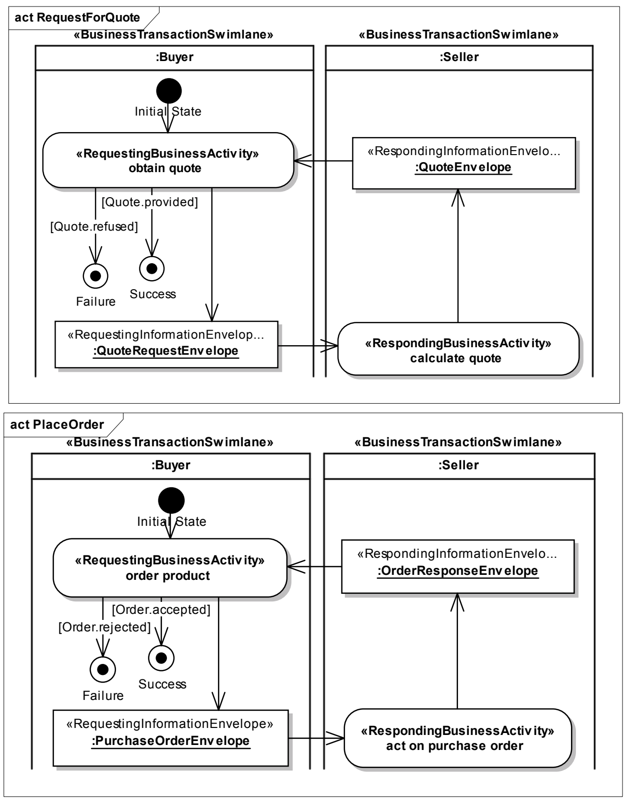

act RequestForQuote
«BusinessTransactionSwimlane»
:Buyer
«BusinessTransactionSwimlane»
:Seller
Initial State
«RequestingBusinessActivity»
obtain quote
«RespondingInformationEnvelo...
:QuoteEnvelope
[Quote.provided]
[Quote.refused]
Failure
Success
«RequestingInformationEnvelop...
:QuoteRequestEnvelope
«RespondingBusinessActivity»
calculate quote
act PlaceOrder
«BusinessTransactionSwimlane»
:Buyer
«BusinessTransactionSwimlane»
:Seller
Initial State
«RequestingBusinessActivity»
order product
«RespondingInformationEnvelo...
:OrderResponseEnvelope
[Order.accepted]
[Order.rejected]
Failure
Success
«RequestingInformationEnvelope»
:PurchaseOrderEnvelope
«RespondingBusinessActivity»
act on purchase order

Fig. 2. Business transactions

In our example, the `request for quote` *business transaction use case* is mapped to the corresponding *business transaction* depicted on top of Fig. 2. It consists of two *business transaction swimlanes*, one for the `buyer`, another one for the `seller`. The `buyer` performs the *requesting business activity* `obtain quote` which outputs a `purchase order envelope` setting the *business entity* `quote` to the interim state `requested`. The envelope is input to the `seller`'s r*equesting business activity* `calculate quote`. The activity sets the final state of the *business entity* `quote` either to `provided` or `refused`. The final state is communicated by returning the `quote response envelope` to the *requesting business activity* `obtain quote`. A *business transaction* follows a strict pattern - only the response is optional. This is easily recognized by a look on the *business transaction* `place order` at the bottom of Fig. 2, which choreographs the requirements of the *business transaction use case* `place order`.

The structural definition of the *requesting/responding business envelopes* exchanged in the *business transactions* is defined in another BTV subview: the *business* *information view*. Since modeling business documents is out of scope for this paper we do not detail them any further. The interested reader is referred to the details given in [15].

The requirements described in the *business collaboration use case* `order from quote` are choreographed in the corresponding activity graph of a *business collaboration protocol*, which is defined in a *business choreography view*. A *business collaboration protocol* specifies a choreography among the *business transactions* defined before. Accordingly, the *business collaboration protocol* `order from quote` - depicted in Fig. 3 - defines a sequence of the *business transaction activities* `request for quote` and `place order`. Each of the two is refined by the corresponding *business transaction* in Fig. 2.

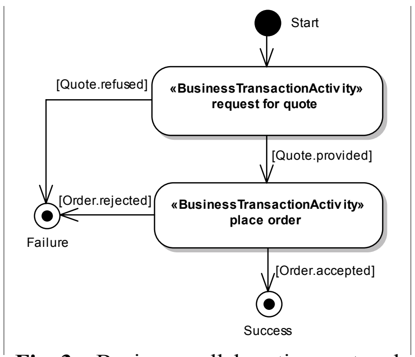


Fig. 3. Business collaboration protocol

3.2 Problem: Nested Business Transaction

In the previous subsection we introduced UMM by means of a bi-lateral collaboration. This means exactly two parties - buyer and seller - collaborate. In this section we take a look how UMM handles multi-party collaborations involving more than two business partners. We continue our previous example, but assume that the seller contacts the buyer's bank to check his credit before giving a quote.

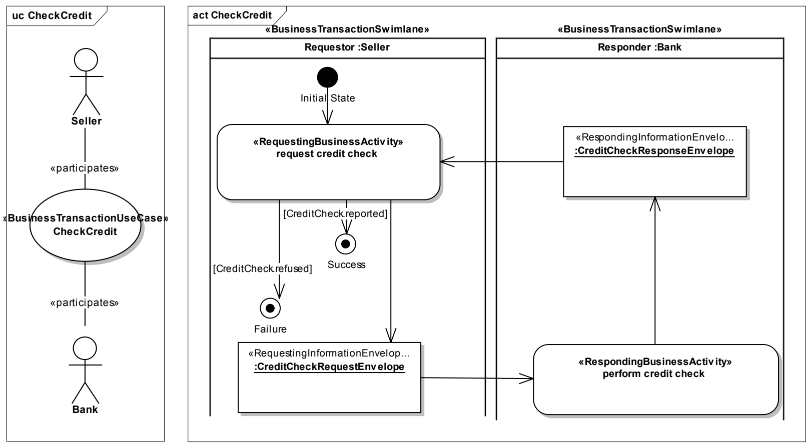

uc CheckCredit
Seller
«participates»
«BusinessTransactionUseCase»
CheckCredit
«participates»
Bank
act CheckCredit
«BusinessTransactionSwimlane»
Requestor :Seller
«BusinessTransactionSwimlane»
Responder :Bank
Initial State
«RequestingBusinessActivity»
request credit check
«RespondingInformationEnvelo...
:CreditCheckResponseEnvelope
[CreditCheck.reported]
Success
[CreditCheck.refused]
Failure
«RequestingInformationEnvelop...
:CreditCheckRequestEnvelope
«RespondingBusinessActivity»
perform credit check

Fig. 4. Check Credit: Business Transaction + Use Case

Evidently, we need another *business transaction use case* that captures the requirements of `check credit`. This use case - happening between the seller and the bank - is depicted at the top of Fig. 4. The bottom shows the corresponding *business transaction* check credit. `Request credit check` is the *requesting business activity* of the `seller` and `perform credit check` is the *responding business activity* of the `bank`. A `credit check request envelope` and a `credit check response envelope` are exchanged between these activities.

We already learned that a *business collaboration protocol* specifies the choreography amongst *business transactions*. The big question is: Is it possible to specify a choreography among *business transactions activities* that may be executed between different pairs of business partners? Thus we check if the *business transaction activity* `check credit` may extend the *business collaboration protocol* `order from quote` as depicted in Fig. 3.

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 buyer's 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* (see Fig. 5). In general, the transition from one *business transaction 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 order to overcome the limitations of nested transactions, one may think to split the encompassing two-way *business transaction* into two one-way *business transactions*: 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 [17].

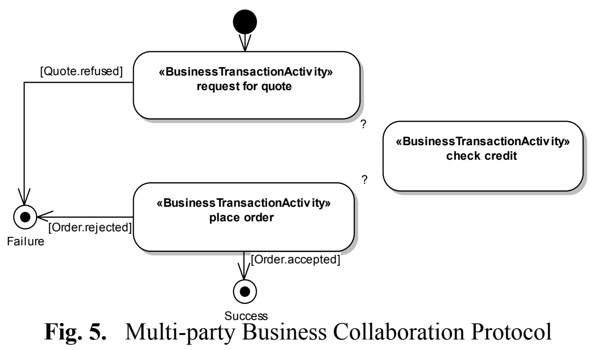

[Quote.refused]
«BusinessTransactionActivity» request for quote
?
«BusinessTransactionActivity» check credit
?
«BusinessTransactionActivity» place order
[Order.rejected]
Failure
[Order.accepted]
Success

Fig. 5. Multi-party Business Collaboration Protocol

Nested *business transactions* do not appear in bi-lateral collaborations. According to the *business transaction* semantics - which are derived from the Open-edi reference model [17] - a responder has first to reply in a two-way *business transaction* before starting another one. Therefore, nested *business transactions* only 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 only suited for bi-lateral collaborations.It follows that a multi-party business collaboration must be split into bi-lateral business collaborations.

4 A UML Profile for Local multi-party Choreographies

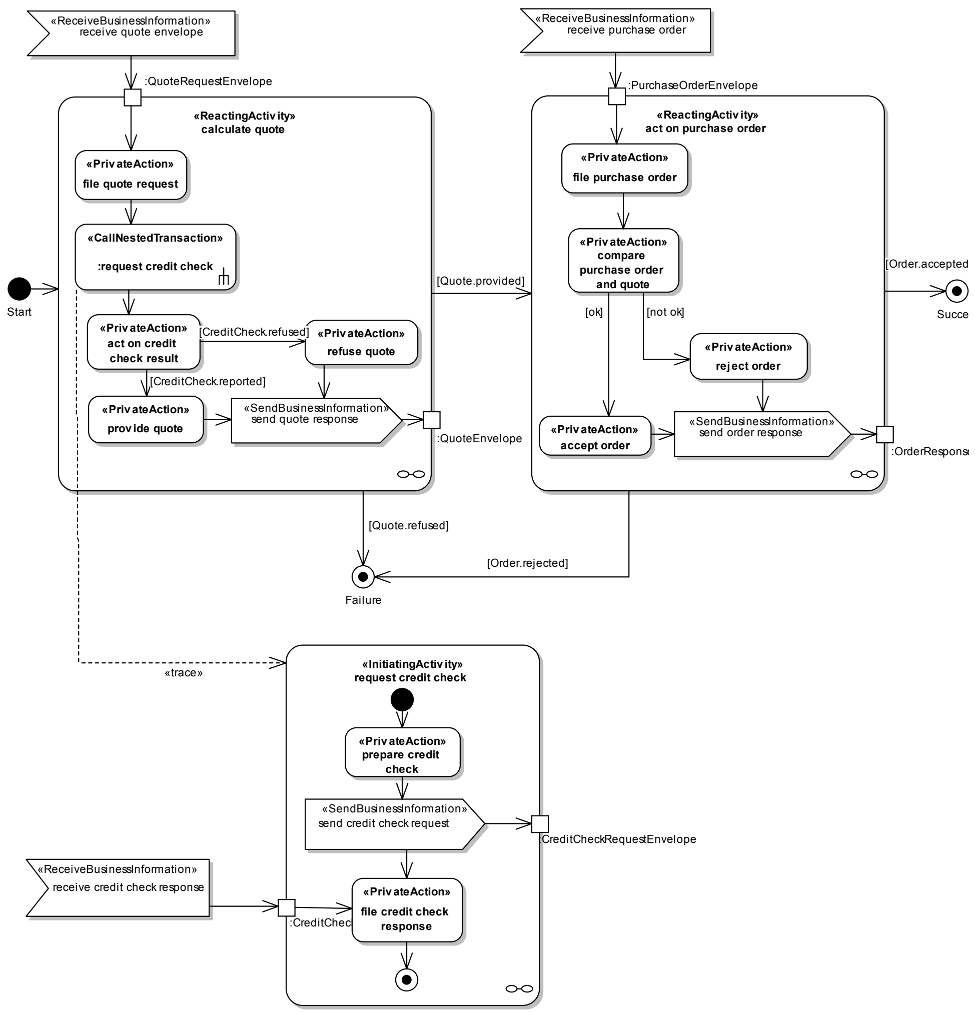
A limitation of splitting up a multi-party business collaboration into multiple bi-lateral business collaborations is the fact, that all dependencies between the different bi-lateral business collaborations are lost. Accordingly, the information that checking the buyer's credit is nested within the request for quote is gone. From a UN/CEFACT perspective this seems to be ok, since UN/CEFACT's 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/CEFACT's 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.

4.1 Steps of specifying a local choreography by example

Again we use the same business case to explain our UML profile for local choreographies, before explaining the meta model later on in section 5.3. We demonstrate the *local choreography* by means of the `seller` in our `order from quote` example. The flow within this *local choreography* is depicted in Fig. 6. Its main building blocks are *initiating activities* and *reacting activities*. An *initiating activity* is used to model the inside of a *requesting business activity*. Similarly an *reacting activity* models the inside of a *responding business activity*. It follows that the `seller`'s *local choreography* includes equivalent *reacting activities* for its *responding business activities* in the transactions of Fig. 2: `calculate quote` and `act on purchase order`.

The next step is to specify the flow between these two *reacting activities*. This flow is derived from the *business collaboration protocol* `order from quote` in Fig. 3. The reacting activities (or the responding business activities respectively) represent the seller's task in the business transaction activities that are choreographed in this business collaboration protocol. It follows that the sequence of `request for quote` and `place order` in Fig. 3 is mapped to a sequence between `calculate quote` and



«ReceiveBusinessInformation» receive quote envelope
:QuoteRequestEnvelope
«ReactingActivity» calculate quote
«PrivateAction» file quote request
«CallNestedTransaction» :request credit check
Start
«PrivateAction» act on credit check result
[CreditCheck.refused]
«PrivateAction» refuse quote
[CreditCheck.reported]
«PrivateAction» provide quote
«SendBusinessInformation» send quote response
:QuoteEnvelope
[Quote.provided]
«ReceiveBusinessInformation» receive purchase order
:PurchaseOrderEnvelope
«ReactingActivity» act on purchase order
«PrivateAction» file purchase order
«PrivateAction» compare purchase order and quote
[ok]
[not ok]
«PrivateAction» reject order
«PrivateAction» accept order
«SendBusinessInformation» send order response
:OrderRespons
[Order.accepted
Succe
[Quote.refused]
[Order.rejected]
Failure
«trace»
«InitiatingActivity» request credit check
«PrivateAction» prepare credit check
«SendBusinessInformation» send credit check request
:CreditCheckRequestEnvelope
«ReceiveBusinessInformation» receive credit check response
:CreditChec
«PrivateAction» file credit check response

Fig. 6. Local Choreography

`act on purchase order` in the seller's local choreography. Furthermore, the guard conditions are mapped. This means, a `refused quote` leads to a failure state. The transition from `calculate quote` to `act on purchase order` is guarded by the fact that a `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 *reacting activities* in the *local choreography*. An object node is added for each incoming and outgoing *information envelope*. *Initiating/reacting activities* taken from two-way transactions have both an input and an output node. If they refer to a one-way *business transaction*, an *initiating activity* has only an output node and a *reacting activity* has only an input node. In our example, `calculate quote` and `act on purchase order` both are derived from two-way *business transactions*. Thus, they have an input and an output node. The *information envelopes* assigned to these nodes correspond to the input and output of the corresponding *requesting/responding business activities*. From Fig. 2 it follows that `calculate quote` has an input of `quote request envelope` and an output of `quote response envelope`. Similarly, `act on purchase order` receives an input of `purchase order envelope` and outputs an `order response envelope`.

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 *initiating/reacting activity*. In case of a *reacting activity* this event and the resulting transfer of the *information envelope* is required to start the *reacting activity*. In our example of Fig. 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`.

In order to demonstrate modeling the flow within an *initiating/reacting activity* we take a look on `calculate quote` in Fig. 6. As mentioned above it is started with an incoming quote request envelope. So the first task of the flow is `file quote request`. `Request credit check` from a bank is the next task. Once this is done one has to `act on the credit check results`. Logically, the next step is either `provide quote` or `refuse quote`. In either case `send quote response` is the last task.

However, it is easy to recognize that these different task modeled as differently stereotyped activities. Most are *private actions* which are tasks internal to the organization. Since these are not visible to others, they are not really part of a local choreography, but of an orchestration. However, first tests have shown that users prefer to model some internal tasks at least to a minimum extent. Thus, we have included *private actions* in our profile - even if we do not focus on orchestration in particular.

The relevant task for a local choreography are `request credit check` and `send quote response`. Thus we have a more detailed look on them. The former is of stereotype *send business information* which is a special kind of the *send signal action.* In our example, `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 `seller`) is not shown in the *local choreography* of Fig. 6 - it is already defined in the `request for quote` *business transaction* on top of Fig. 2.

Check credit is of stereotype *call nested transaction* which addresses the problem described in section 3.2. As we know from our example, checking the customer credit has do 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 seller's `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 an *initiating/reacting activity* of the same party. In our example of Fig. 6, the `calculate quote` activity includes the *call nested transaction* `request credit check`. This one calls the synonymously named *initiating activity* `request credit check` - which is the `sellers` task in the *business transaction* `check credit` (see Fig. 4).

We again specify a flow within the *initiating activity* `request credit check`. After finishing this flow, control is given back to `calculate quote`. Since `request credit check` is an *initiating 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 *initiating/reacting activity*. If a whole collaboration is nested, the *call nested collaboration* concept is used. This one calls another *local choreography*.

5 UML profile definitions

We introduced in section 3 the UMM and in section 4 our proposed UML profile for local choreographies - both by the means of an example. These examples are based on an underlying meta model that is defined as a UML profile. A UML profile customizes the general purpose modeling language UML for a specific purpose - in our case modeling inter-organizational business systems. It comprises a set of stereotypes, their tagged value definitions, and OCL constraints. Usually, stereotypes have some conceptual relationships with each other. However, the definition of these relationships is not directly part of a UML profile. Since the UML meta model is used, these relationships must be reflected in the existing relationships of the UML meta model - this is realized by the OCL constraints of a UML profile. In this section we introduce the relevant parts of the UMM meta model and the one of our profile for local choreographies. For a better readability and understanding, we show a conceptual model of the relationships between the stereotypes instead of the OCL constraints.

5.1 Stereotypes of business collaboration protocols

A *business choreography view* is used to define exactly one *business collaboration protocol*. The bu*siness collaboration protocol* follows exactly the requirements defined in a corresponding *business collaboration use case* of the BRV. The activities of a *busi-*

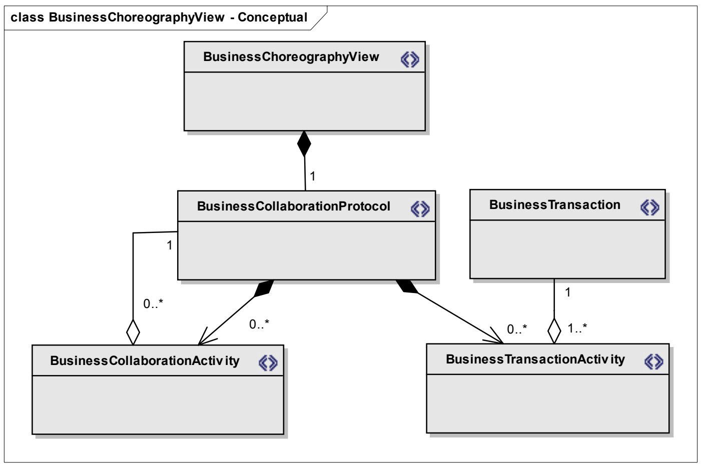

class BusinessChoreographyView - Conceptual
BusinessChoreographyView
1
BusinessCollaborationProtocol
BusinessTransaction
1
0..*
0..*
1
0..*
1..*
BusinessCollaborationActivity
BusinessTransactionActivity

Fig. 7. Business Collaboration Protocols: Conceptual Model

ness collaboration protocol are *business collaboration activities* and/or *business transaction activities*. Hence, a *business collaboration protocol* is composed of zero to many *business collaboration activities* and of zero to many *business transaction activities*. However, at least one *business collaboration activity* or a *business transaction activity* must be present in a *business collaboration protocol*. Transitions defining the flow among the *business collaboration activities* and/or *business transaction activities* may be guarded by the states of business entities.

A *business collaboration activity* is characterized by the fact that it is refined by another *business collaboration protocol*. Not each business collaboration is a refined *business collaboration activity* - only the nested *business collaboration protocols*. A *business collaboration protocol* may be nested in different *business collaboration activities*.

A *business transaction activity* is characterized by the fact that it is refined by a *business transaction*. Since the *business transaction* is a concept of the *business interaction view* it is described in more detail further below. Each *business transaction* must be at least once used to refine a *business transaction activity*. A *business transaction* may be nested in different *business transaction activities*.

5.2 Stereotypes of business transactions

A *business interaction view* comprises exactly one *business transaction* synchronizing the entity states between the two *authorized roles*. Each *business transaction use case* of the *business requirements view* is mapped to exactly one *business transaction*.

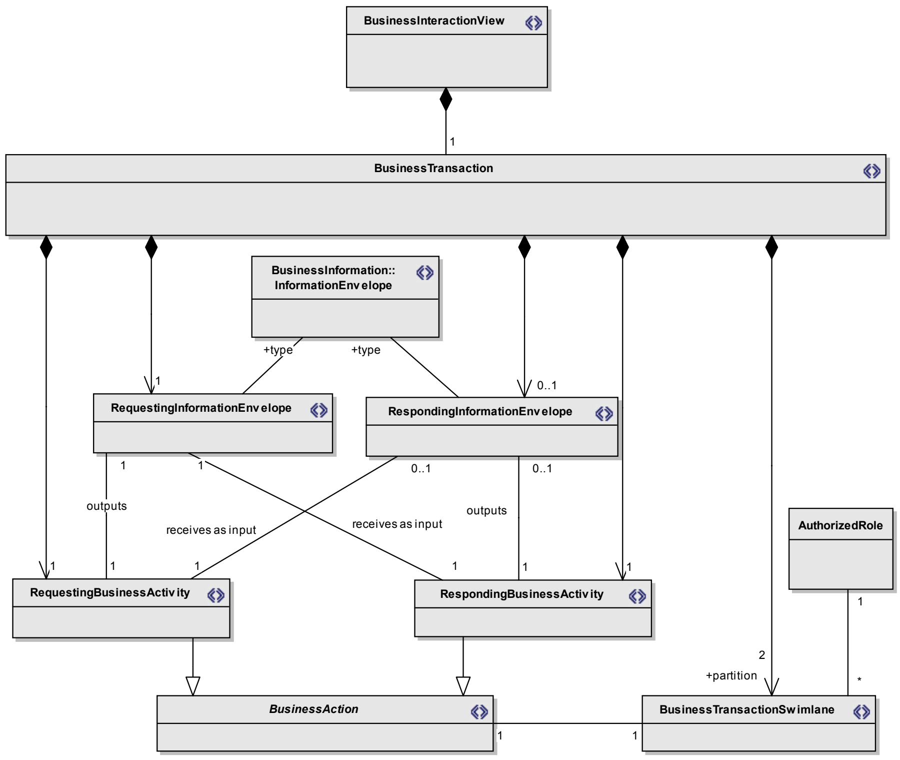


Fig. 8. Business Transactions: Conceptual Model

A *business transaction* is an atomic business process between two *authorized roles*, which involves sending *business information* from one *authorized role* to the other and an optional reply. The *business transaction* is built by two partitions - one for each *authorized role*. Hence, a *business transaction* is composed of exactly two *business transaction swimlanes*. Each *business transaction swimlane* relates to one *authorized role*. An *authorized role* is assigned to exactly one *business transaction swimlane*. It follows, that the two swimlanes of a *business transaction* must be assigned to different *authorized roles*.

Within a *business transaction* each *authorized role* performs exactly one *business action* - the requesting *authorized role* performs a *requesting business activity* and the responding *authorized role* performs a *responding business activity*. Each *business action* - no matter whether *requesting* or *responding business activity* - is assigned to a *business transaction swimlane*, and each *business transaction swimlane* comprises exactly one *business action*.

The *requesting business activity* outputs the *requesting information envelope* that is input to the *responding business activity*. The *responding information envelope* created by the *responding business activity* and returned to the *requesting business activity* is optional. Note, that a *requesting information envelope* (or a *responding information envelope*) is a stereotype of the base class object flow state. The type of the object flow state is defined by the *information envelope* that is a stereotype of base class class.

The stereotype of a *business transaction* comprises two tagged values. The tagged value *is secure transport required* is used to indicate that the exchange of business information must use a secure transport channel. The business transaction type is an enumeration of six different types: *Information distribution* indicates an informal information exchange whereas *notification* signals a formal information exchange according to an underlying contract. Both types are one-way exchanges without an response. The following types require a response: In *query/response* the responder has the information available before the request is made, in *request/response* this is not the case. *Request/confirm* requires only confirmation of a request with respect to previously established contracts. A *commercial transaction* results in a residual obligation between the partners.

The different types of *business transactions* also differ in the default values of the tagged values for *business actions*: *time to acknowledge receipt*, *time to acknowledge acceptance*, *is authorization required*, *is non repudiation required*, *is non repudiation of receipt required*, and *is intelligible check required*. These tagged values of *business action* are inherited by the *requesting business activity* and the *responding business activity*. In addition the *requesting business activity* specifies the tagged values of *time to respond* and *retry count*. Most of these tagged values are self-explanatory. An *acknowledgment of receipt* is usually sent after grammar validation, sequence validation, and schema validation. However, if *is intelligible check* is set to false, this acknowledgment is sent without any of the validations. An *acknowledgment of acceptance* is sent after performing a validation of additional business rules that are required by the target business application. *Retry count* is the number of retries initiated by the requestor in case of control failures.

An *information envelope* is characterized by three security parameters: *is confidential*, *is tamper proof* and *is authenticated*. These self-explanatory tagged values are inherited from information entity.

5.3 Stereotypes of local choreographies

As already mentioned earlier, a choreography describes the flow of interactions between the participating business partners that interlink their individual processes. A local choreography describes the flow from a participating partner's point. In this section we introduce our meta model for local choreographies which offers a solution to model nested business collaborations. Our proposal includes the following set of stereotypes that have already been used in the example of section 4:

- Local Choreography View
- Local Choreography
- Local Activity - which is an abstract superclass for
  - Initiating Activity
  - Reacting Activity
- Receive Business Information
- Send Business Information
- Call Nested Transaction
- Call Nested Collaboration
- Private Action
- Private Activity

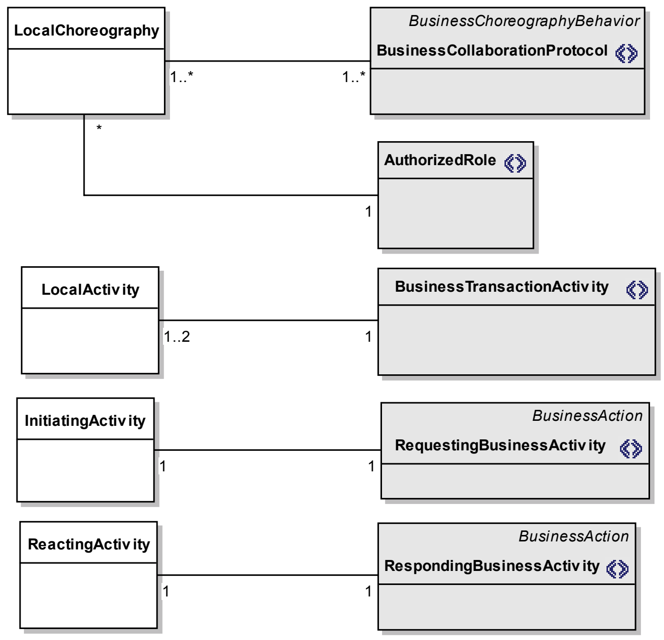

LocalChoreography
BusinessChoreographyBehavior
BusinessCollaborationProtocol
1..*
1..*
*
AuthorizedRole
1
LocalActivity
BusinessTransactionActivity
1..2
1
InitiatingActivity
BusinessAction
RequestingBusinessActivity
1
1
ReactingActivity
BusinessAction
RespondingBusinessActivity
1
1

Fig. 9. Relationships between Stereotypes

We already mentioned in section 1 that the justification for developing another choreography language is the dedicated binding to UMM allowing a straight-through modeling approach. Accordingly, the stereotypes of our local choreography have well-defined relationships to the UMM stereotypes. These relationships are depicted in Fig. 9.

A *local choreography* describes a flow of activities from a participating party's point of view. It extends the concept of a UML activity, that is itself composed of further activities. All its activities are performed by the same party. Consequently, a *local choreography* is assigned to exactly one *authorized role* from UMM. Of course, an *authorized role* may perform many different *local choreographies* representing many different business cases.

A UMM *business collaboration protocol* always involves two parties. A *local choreography* - although executed by a single party - involves interactions with multiple parties, i.e. interactions defined in different *business collaboration protocols*. It follows, that a l*ocal choreography* is related at least to one, but up to many *business collaboration protocols*. A *business collaboration protocol* may be reflected in many different *local choreographies*.

When a UMM *business collaboration protocol* is part of a *local choreography*, it is mandatory that each of its *business transaction activities* results in a *local activity*, or in other words in an *initiating activity* or in a *reacting activity*. Since a *business transaction activity* is performed by two parties, it may be the source of up to two *local activities*. However, it should be noted that these two *local activities* will never be used in the same *local choreography*, because they are not executed by the same *authorized role*. Contrariwise, each *local activity* is backed up by exactly one *business transaction activity*.

In UMM, a *business transaction activity* is refined by a *business transaction*, in which the *authorized role* under consideration in the *local choreography* performs either a *requesting business activity* or a *responding business activity*- which one is defined by the *authorized role* assigned to the *business transaction swimlane* hosting the *requesting/responding business activity*. If the *authorized role* executes a *requesting business activity*, the *local choreography* includes an *initiating activity*. In case of a *responding business activity*, the *local choreography* comprises a *reacting activity*. In other words a *requesting business activity* and *an initiating activity* (as well as a *responding business activity* and a *reacting activity*) represent they same "logical" concept - however one is used in the flow of inter-organizational systems and the other one in the local flow. In order to not mix up the flows the "logical" concepts results in two different stereotypes. However, there is always a one-to-one relationship between *requesting business activity* and *initiating activity* as well as between *responding business activity* and *reacting activity*.

Fig. 10 shows the relationships between the stereotypes of the UML profile for local choreographies. It is our intension that local choreographies may be specified in the same model as the related UMM model. However, all artifacts of the local choreography must reside in their own package structure independent of the UMM. Thus, the stereotype *local choreography view* is a new type of a top level package corresponding model. A local choreography view may cover many local choreographies.

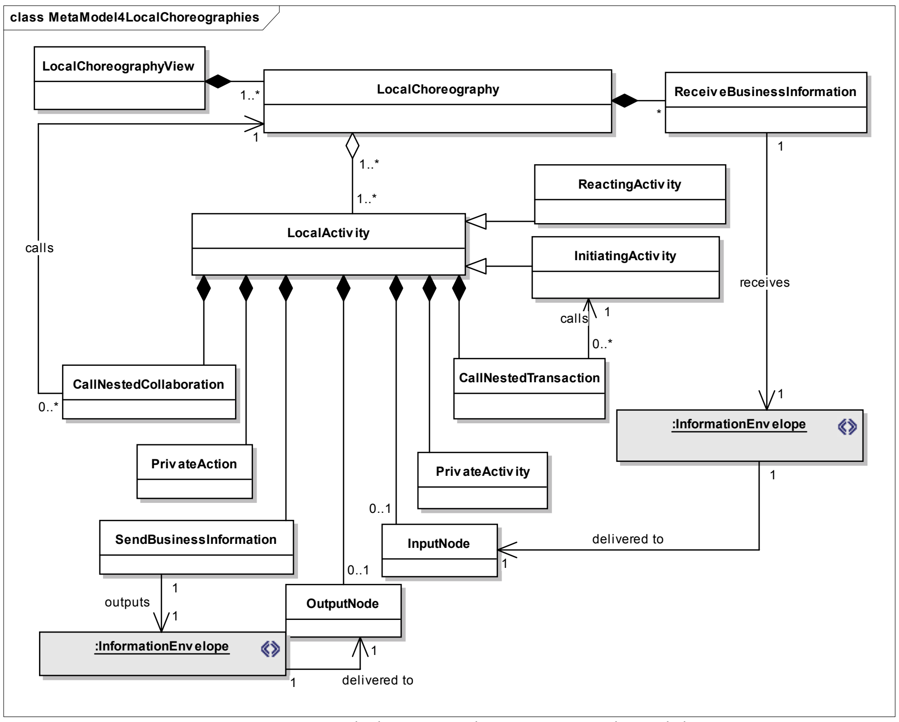


Fig. 10. Local choreography: Conceptual Model

A *local choreography* includes at least one, but up to many *local activities* - no matter whether they are *initiating activities* and/or *reacting activities*. However, a *local choreography* does not exclusively own a *local activity*. One and the same *local activity* may be re-used in different *local choreographies* - since the related *business collaboration protocol* may be used in different *local choreographies*. It is important to note that the transitions between *local activities* must correspond - including their guards - to the transitions in the *business collaboration protocol* from which the *local business activities* are derived.

A *local choreography* may include *receive business information* activities in addition to *local activities* - however no other children are allowed. The concept of the stereotype *receive business information* is explained below.

In UMM, requesting/responding business activities send and/or receive information envelopes. This fact must be reflected in local activities as well. Thus, an object node is added to a *local activity* for each incoming and outgoing *information envelope*. *Local activities* taken from two-way transactions have both an *input node* and *an output node*. If they are part of a one-way *business transaction*, an *initiating activity* has only an *output node* and a *reacting activity* has only an *input node*.

For each input node we add a *receive business information* - which is a UML *accept event action* - to the *local choreography*. It is used to recognize the event of an incoming *information envelope* and to deliver it to the input node of the corresponding *local activity*. In case of a *reacting activity* this event and the resulting transfer of the *information envelope* is required to start the *initiating activity*.

The flow within a local *activity* comprises 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 deliver an *information envelope* to the *output node* of the *local activity*. It should be mentioned that output nodes of a *local activity* do not show any further object flow in the *local choreography*. This means the flow of sending the information envelope to the other party must be specified in a UMM *business transaction*.

The stereotype *call nested transaction* is another kind of action in the flow of a *local activity*. It is as a special kind of the UML *call action behavior* used to call a flow in another *local activity* of local choreography. However, it should be noticed that *call nested transaction* is only used for calling a single *initiating/reacting activity*. If a whole collaboration is nested, the *call nested collaboration* concept is used. This one calls another *local choreography*.

6 Summary

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 (cf. http://www.unece.org/cefact/brs/brs_index.htm), as well as by national e-government frameworks, such as XÖV 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 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. For these reasons we provide a methodology extending the UMM for the purpose of modeling local choreographies as well as multi-party collaborations. Our approach extends the UMM by another set of stereotypes and related constraints on the meta model. In this paper we define the stereotypes and their relationships with UMM. The approach specified by our profile guarantees that the local choreography respects the commitments made in the global choreography - assuming that the global choreography is correct. Furthermore, we demonstrate the methodological steps 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 [13]. 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.

7 References

1. 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
2. van der Aalst, W.M.P., Weske, M.: The P2P approach to Interorganizational Workflows. Proceedings of the 13th International Conference on Advanced Information Systems Engineering (CAISE), Springer LNCS (2001) 140-156
3. van der Aalst, W.M.P.: Inheritance of Interorganizational Workflows to Enable Business-to-Business. Journal of Electronic Commerce Research,Vol. 2, No. 3, (2002) 195-231
4. 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
5. Barros, A., Dumas, M., and Oaks, P. Standards for Web Service Choreography and Orches-tration: Status and Perspectives. In Workshop on Web Service Choreography and Orches-tration for Business Process Management @ BPM 2005, Springer LNCS (2005)
6. Booch, G., Rumbaugh J., Jacobson, I.: The Unified Modeling Language User Guide, Second Edition. Addison Wesley, Reading (2005)
7. Decker, G., Kopp, O., Leymann, F., Weske, M.: BPEL4chor: Extending BPEL for Modeling Choreographies. Proc. of the IEEE 2007 Int'l Conf. on Web Services (ICWS), IEEE CS, (2007)
8. Geerts, G. and W.E. McCarthy. Modeling business enterprises as value added process hierarchies with resource-event-agent object templates. Business Object Design and Implementation: OOPSLA'95 Workshop Proceedings, Springer (1997)
9. Gordijn, J., Akkermans, H.: E3-value: Design and Evaluation of e-Business Models.

IEEE Intelligent Systems, Vol. 16, No. 4, (2001) 11 - 17

10. Hofreiter B., Huemer, C., Kim J.-H: Choreography of ebXML business collaborations. Journal of Information Systems and e-Business, Vol. 4, No. 3, Springer (2005) 221-243
11. Hofreiter, B., Huemer, C., Liegl, P., Schuster, R., Zapletal, M.: UN/CEFACT's Modeling Methodology (UMM): A UML Profile for B2B e-Commerce. Conceptual Modeling - ER 2006 Workshops, Spinger LNCS, (2006) 19-31
12. Hofreiter, B., Huemer, C., Liegl, P., Schuster, R., Zapletal, M.: Deriving executable BPEL from UMM Business Transactions. Proc. of the 2007 IEEE International Conference on Services Computing (SCC), IEEE CS, (2007)
13. Hofreiter, B., Huemer, C., Liegl, P., Schuster, R., Zapletal, M.: UMM Add-In: A UML Extension for UN/CEFACT's Modeling Methodology. To appear in: Proc. of 5th Int'l Conf. on Service Oriented Computing, Springer LNCS, (2007)
14. Huemer, C.: UN/CEFACT's Modeling Methodology (UMM): UMM Meta Model - Foundation Module Version 1.0 Technical Specification, UN/CEFACT TMG, (2006) http://www.unece.org/cefact/umm/UMM_Foundation_Module.pdf
15. Huemer, C., Liegl, P.: A UML Profile for Core Components and their Transformation to XSD. 2nd Int'l Workshop on Services Engineering (SEIW 2007) - Proceedings of IEEE ICDE Workshops, IEEE Computer Society, (2007)
16. Huemer, C., Schmidt, A., Werthner, H., Zapletal, M.: Mapping e3Value to UML. submitted to to the 23rd Annual ACM Symposium on Applied Computing (ACM SAC 2008)
17. ISO: Open-edi Reference Model. ISO/IEC JTC 1/SC30 ISO Standard 14662 (1997)
18. Kavantzas, N. et al: Web Services Choreography Description Language, Version 1.0. W3C (2004) http://www.w3.org/TR/ws-cdl-10
19. Khalaf R.: From RosettaNet PIPs to BPEL processes: A three level approach for business protocols. Data & Knowledge Engineering, Vol. 61, No. 2, Elsevier, (2007)
20. Korherr, B., List, B.: Extending the UML 2 Activity Diagram with Business Process Goals and Performance Measures and the Mapping to BPEL. In: Proceeding of the ER-Conference on Conceptual Modeling (Workshop BP-UML'06), Springer (2006)
21. Kramler, G., Kapsammer, E., Kappel, G., Retschitzegger, W.: 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). (2005)
22. 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
23. Lenz, K., Oberweis, A.: Interorganizational Business Process Management with XML Nets. In: Advances in Petri Nets. Springer LNCS, Vol. 2472 (2003)
24. Leymann, F., Roller, D., Schmidt, M.-T.: Web Services and Business Process Management. IBM Systems Journal, Vol. 41, No. 2, 2002
25. 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)
26. List, B., Korherr, B.: A UML 2 Profile for Business Process Modelling. In: ER 2005 Workshop Proceedings. (2005)
27. Osterwalder, A., Pigneur, Y.: An e-Business Model Ontology for Modeling e-Business, 15th Bled E-Commerce Conference, 2002

28. Peltz, C.: Web Services Orchestration and Choreography. IEEE Computer, Vol. 36, No. 10, IEEE Computer Society, (2003) 46-52
29. Penker, M., Eriksson, H.E.: Business Modeling With UML: Business Patterns at Work. Wiley (2000)
30. Piccinelli, G., Emmerich, W., Zirpins, C., Schütt, K.: Web Service Interfaces for Inter-Organisational Business Processes: An Infrastructure for Automated Reconciliation. 6th Int'l Enterprise Distributed Object Computing Conf. (EDOC), IEEE Computer Society, (2002), 285-292
31. RosettaNet: RosettaNet Implementation Framework, Core Specification V02.00.01. (2002) http://www.rosettanet.org/rnif
32. Schatz W.: EDI: Putting The Muscle In Commerce And Industry. Datamation, Vol. 34, No. 6, (1988)
33. UN/CEFACT: ebXML - Business Process Specification Schema v1.10. (2003) http://www.untmg.org/downloads/General/approved/ebBPSS-v1pt10.zip
34. UN/CEFACT: Core Components Technical Specification, Version 2.01. (2003) http://www.unece.org/cefact/ebxml/CCTS_V2-01_Final.pdf
35. White, S: Business Process Modeling Notation (BPMN) Specification , OMG Final Adopted Specification dtc/06-02-01 (2006), http://www.omg.org/docs/dtc/06-02-01.pdf http://www.bpmi.org/bpmn-spec.esp