Visualizing B2Bi Choreographies

Andreas Schönberger
Distributed Systems Group
University of Bamberg
Bamberg, Germany
andreas.schoenberger@uni-bamberg.de

***Abstract*—The implementation of cross-organizational processes in the context of Business-to-Business integration (B2Bi) requires extensive communication between all interacting parties and adequate solutions for the integration of existing IT systems. In the SOA domain, choreography languages have been put forward for the specification of the publicly visible message exchange sequences of interacting parties. Choreography models serve as basis for communication between stakeholders and as reference for aligning business services which is a core SOA concept.**

However, visual choreography modeling guidelines that are both, tailored to B2Bi and unambiguous in semantics, have not been proposed so far. This paper fills in this gap by showing how BPMN choreography notation can be used to represent so-called B2Bi choreographies with unambiguous semantics. The work has been validated by a series of real-world use cases of the RosettaNet community and has been accepted by the RosettaNet Message Control and Choreography team as core part of a RosettaNet standard.

***Keywords*-B2Bi; Choreography; business services; BPMN; ebXML BPSS**

I. Introduction

The design and implementation of Business-to-Business integration (B2Bi) processes is challenging in organizational ways as much as in technical ways. First, process modelers of the participating enterprises have to identify the business messages to be exchanged and the admissible exchange sequences on an abstract level. The process modelers' design of business message exchanges then has to be communicated to software engineers of all parties who turn the input design into business services that perform the actual message exchanges. Thereby, the implementation process must deal with challenging distributed systems problems such as heterogeneity and transaction safety. However, this transformation must not challenge strict conformance of business service implementations to process models.

SOA practice suggests the use of choreography models for specifying the publicly visible message exchange sequences between the interacting parties of cross-organizational business processes. In [1], B2Bi choreographies, service choreographies and conceptual choreographies are identified as distinct types of choreography languages. While B2Bi choreographies are semantically close to business process models and use business document exchanges as core concept, service choreographies use services to define implementation-centric message exchange specifications. Conceptual choreographies, in turn, use formalisms such as Petri nets for analysis purposes.

In [2], the current state of cross-organizational business process implementations is analyzed and the need for *"stringent, vendor-independent business process standards"* that enable *"quick establishment of IS integration"* is postulated. In the area of B2Bi choreographies, approaches based on the ebXML Business Process Specification Schema (ebBP, [3]) such as [4] or [5] provide exactly that. The basic concept is to formally define B2Bi choreography classes with unambiguous semantics that are known to be valid in the sense of being transformable into BPEL-based implementations. However, as ebBP is a verbose XML-based standard, the communication function of choreography models is not supported well. Current *visual choreography standards*, in particular BPMN choreographies [6, section 11] and UMM [7], are candidates for filling this gap, but these do not formally define valid choreography classes (BPMN, UMM) or are not tailored to B2Bi (BPMN).

The contribution of this work is an adaptation of BPMN choreographies to the purpose of B2Bi choreography modeling based on established B2Bi choreography modeling approaches [4], [5], [8]. The adaptation provides a visual representation of B2Bi choreographies that is easy to use and abstracts from technical detail, but nonetheless is precise in semantics. For validation, 11 use cases of the RosettaNet B2Bi community have been modeled using the BPMN-based B2Bi choreography visualization. The visualization also has been accepted by the RosettaNet Message Control and Choreography team as core part of the community standard with the title *RosettaNet Methodology for Creating Choreographies* [9].

The paper proceeds as follows: Section II introduces the most important B2Bi terms for the purpose of this work and section III describes the methodical foundation for restricting BPMN. Section IV introduces the BPMN elements used for B2Bi choreography modeling and section V exemplifies the most important rules for building valid choreographies from these elements. BPMN compliance as well as the validation of the visualization is discussed in section VI and section VII presents related work. Conclusion and future work are given in section VIII.

II. Basics

For the purpose of this paper, some terminology taken from dedicated B2Bi choreography standards is needed.

The atomic building block of B2Bi choreographies are so-called BusinessTransactions (BTs) as defined in ebBP [3] or UMM [7]. A conceptually equivalent concept is defined by RosettaNet's Partner Interface Processes (PIPs)[1] which are the basis for the examples in the next sections. A BT defines the exchange of a request and an optional response business document such as quotes, orders or invoices between a so-called requester role and a responder role. The sender of the request document is assigned the requester role whereas the receiver is assigned the responder role. The execution of BTs is defined in full-fledged exchange protocols [10], [11] that include security and reliability configurations as well as the exchange of processing signals that indicate the legibility or validity of business documents. By means of these protocols, the result of a BT is synchronized between the participating roles and generic protocol outcomes such as *ProtocolSuccess* or *AnyProtocolFailure* are computed. BusinessCollaborations (BCs; also referred to as B2Bi choreography) define compositions of BT executions by means of choreographing so-called BusinessTransactionActivities (BTAs). Thereby, a BTA maps the roles of a BC to the requester/responder role of the BT under execution. By analogy, BusinessCollaborationActivities can be used to specify the execution of BCs within other BCs. The control flow of BCs is based on constructs such as forks, joins, or decisions as well as on expression languages such as *ConditionGuardValue* (CGV; defined in ebBP) for referring to generic protocol outcomes of BTAs or XPath for referring to the exchanged contents.

III. Approach

The goal of coming up with a notation for B2Bi choreography modeling that is simple and abstract on the one hand and yet unambiguous in semantics on the other has driven the approach for identifying the BPMN elements used for B2Bi choreography modeling and for assigning B2Bi specific meaning. In essence, the visualization in the next two sections is a simplification of BPMN choreographies for the purpose of B2Bi choreography modeling and deliberately is not strictly compliant to BPMN. The methodical foundation for restricting BPMN is given by [12] who identify four main grammatical deficiencies of languages compared to the ontological domain these are applied to:

- *"**Construct overload**: Several ontological constructs map to one grammatical construct.*
- ***Construct redundancy**: Several grammatical constructs map to one ontological construct.*
- ***Construct excess**: A grammatical construct might not map to any ontological construct.*
- ***Construct deficit**: An ontological construct might not map to any grammatical construct."*

While the BPMN choreography notation provides the grammatical constructs for a corresponding analysis the ontological domain for choreography modeling cannot be identified as easily and therefore is narrowed down as follows.

The ebBP-ST [4], ebBP-Reg [5] and SeqMP [8] B2Bi choreography classes all reflect the results of an extensive literature study on B2Bi requirements [13]. This requirements study, in turn, is based on B2Bi standards such as UMM [7] or ebBP [3] and B2Bi literature. Therefore, ebBP-ST, ebBP-Reg and SeqMP can be considered to be relevant for B2Bi. In a second step, 100 processes derived from RosettaNet's *RosettaNet Implementation Guide* (RIG) library have been analyzed for required choreography features. *"[A RIG] describes the specific business scenario(s), usage notes and lessons learned [when implementing PIPs]"* which is supposed to *"help reduce implementation time and accelerate adoption of the process scenario by sharing the experience of early implementers."*[2] Control flow requirements of RIGs can be deduced from the business scenario descriptions of RIGs and the overwhelming majority of RIGs is pretty simple in that regard. Only 44 out of 100 RIGs use hierarchical decomposition, 15 RIGs describe multi-party scenarios, 12 RIGs use loops, and 8 RIGs have parallel activities. In consequence, the above B2Bi choreography classes cover the scenarios of the RIG library pretty well. Following the *satisficing* principle of Simon [14], mapping ebBP-ST, ebBP-Reg and SeqMP to BPMN hence promises good coverage for visualizing B2Bi choreographies.

The inadequacy of *raw* BPMN choreographies for the purpose of modeling such B2Bi choreographies is reflected in *construct deficits* and *construct excesses*. On the one hand, B2Bi domain concepts such as BTs are missing (see BPMN extensions described in section VI) while a significant number of BPMN choreography concepts is not required for capturing B2Bi choreographies on the other (compare the BPMN constructs used below with the constructs offered by the BPMN standard).

The next sections describe how these B2Bi choreography classes can be captured using BPMN choreographies. Thereby, adequate representation of B2Bi concepts is valued higher than strict BPMN compliance. So-called *executable choreographies* are defined as visual superset of ebBP-ST and ebBP-Reg whereas *SeqMP choreographies* have a correspondingly named visualization style. Both visualization options got accepted by RosettaNet's *Message Control and Choreography* team[3] as core part of the community standard

[1] http://www.rosettanet.org

[2] http://www.rosettanet.org/Support/ImplementingRosettaNetStandards/RosettaNetImplementationGuides/tabid/2985/Default.aspx

[3] http://www.rosettanet.org/dnn_rose/Standards/RosettaNetPrograms/FoundationalPrograms/ActiveFoundationalPrograms/MessageControlChoreography/tabid/3096/Default.aspx

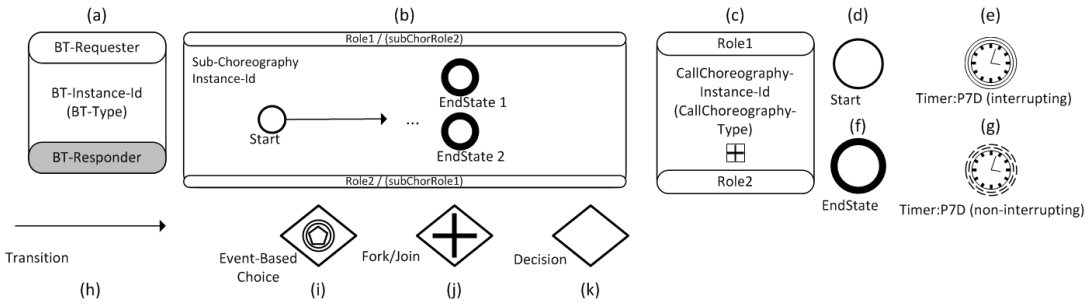

Figure 1. Constructs for B2Bi Choreography Modeling

RosettaNet Methodology for Creating Choreographies [9].

IV. Modeling Elements

This section presents the BPMN elements that are used for B2Bi choreography modeling and the meaning that is assigned. In the subsequent section, valid connections between these elements are presented and section VI discusses compliance to the BPMN standard.

The 11 BPMN constructs needed for representing executable choreographies and SeqMP choreographies are shown in figure 1. The first three elements (a)-(c) represent the actual interaction activities (BTAs, BCAs) whereas all other elements are used to define control flow.

Figure 1 (a) shows a so-called BPMN choreography task (rounded rectangle with two bands) that is used to represent a BTA. Remember that a BTA is the atomic building block of a choreography that produces a result which is synchronized between the two participating roles according to a full-fledged execution protocol (cf. section II). In the middle of the BTA shape, the activity id is given together with the BT type in parentheses. While the BT type identifies the business documents to be exchanged and a standard assignment of execution protocol parameters, the activity id is used to distinguish between multiple executions of the same BT type. The bands at the top and at the bottom of the BTA shape are used to associate the choreography roles with the BT requester and BT responder role where the white band identifies the BT requester and the gray band identifies the BT responder (the position of the band, top or bottom, is insignificant).

Figure 1 (b) is a so-called expanded BPMN sub-choreography that is used for representing BCAs. Again two bands at the top and at the bottom are used to associate the choreography roles with the roles of the underlying BC. However, as the roles of BCs are not predefined (such as requester and responder role for BTs) the mapping is done by means of names where the BC role name is given in parentheses after a slash character. This explicit mapping only can be omitted if the choreography role names coincide with the BC role names. Note that both bands are deliberately kept in white because BCAs may be initiated by both participating roles. In the upper-left corner of the middle band, the activity id of the BCA is given. The BC type is defined anonymously by means of the shapes contained inside of the middle band which define the actual sub-choreography.

The shape of figure 1 (c) is a so-called collapsed BPMN call choreography that also is used for representing BCAs. In this work, it has basically the same meaning as shape 1 (b). The difference is that the BC to be performed is defined in a separate choreography and therefore the activity in the middle of the shape is accompanied with the name of the BC to be performed in parentheses. Again, the two participant bands can be used for associating the choreography roles with the BC roles. Only defining the choreography roles expresses the assumption that the BC uses the same role names as the superordinate choreography. Mapping of roles then can be performed by checking for string equality.

The shapes of figure 1 (d) and (f) show start and end states, respectively, that are used for demarcating choreographies, and the shapes of figure (e) and (g) represent interrupting and non-interrupting timers where the event for firing a timeout can be determined by means of an ISO 8601 compliant time specification (“P7D” denotes Period of 7 Days). Non-interrupting timers can be used for specifying a timeout event for BCAs and for event-based choices (figure 1 (i)) while interrupting timers only can be specified for BCAs (for the effect of an interrupt on the currently executed activity please see [9]).

Shape 1 (h) shows an unconditional transition that can be used to connect the other shapes. Transitions can carry guards where the set of eligible expression languages is imported from the ebBP standard, most notably the CGV and the XPath language (cf. section II). Additionally, the expression languages CBRes and Par have been invented (in compliance with the ebBP and BPMN standard which both allow for introducing new expression languages; cf. [3, section 3.4.11.1.1] and [6, section 10.3]) for capturing the result of BCAs and parallel structures, respectively.

Finally, the shapes of figure 1 (i), (j) and (k) show event-based choice nodes, fork/join nodes and decision nodes, respectively. Event-based choices select between concurrent

events where starting the next BTA, BCA or a timeout are eligible events. Forks and joins are used to demarcate parallel structures where the branches of parallel structures are not allowed to overlap. Decisions can be used to bundle the outgoing transitions of a BTA, BCA or parallel structure. Strictly speaking, decisions are just syntactic sugar because guarded transitions could be attached to BTAs, BCAs and the join node of parallel structures as well.

V. Grammar Rules

The definition of valid *executable choreographies* just requires 12 grammar rules and the definition of valid *SeqMP choreographies* only requires 10 grammar rules (cf. [9]). However, enumerating all rules still exceeds the space limitations of this paper so that the sample processes of figures 2 and 3 are used to exemplify the different options for connecting the modeling elements of the last section. Note that these rules are sufficient for capturing the processes of the underlying RosettaNet RIG analysis.

Figure 2 shows a use case composed of several RosettaNet PIPs ranging from the exchange of a quote (PIP 3A15) to the exchange of a certificate of analysis (PIP 2A17 and PIP 2A18) and covers the most important control flow features of executable choreographies. The underlying paradigm of executable choreographies is a state machine where the BTAs, BCAs and control flow nodes are interpreted as states. For example, the BTA 'PIP 3A15' with activity id 'act-id-1' is interpreted as state as well as the BCA with activity id 'subchor-1', the event-based choice node 'Pending' at the bottom of the figure or all the top-level start and end states. Note that the fork-join pair in the middle of figure 2 is interpreted as one single state as well. The sample executable choreography does not include a collapsed call choreography shape, but BCAs 'subchor-1' and 'subchor-2' could be replaced by such shapes and the contained BCAs could be defined in a separate choreography definition. For end states, BTAs and BCAs, names or activity ids are required while names may be assigned to all other states on a non-obligatory basis. These names must be unique relative to the enclosing choreography. Therefore, assigning 'act-id-1' to the BTA 'PIP 3A15' at the start and to the BTA 'PIP 3A19' in BCA 'subchor-1' is perfectly acceptable.

For the interpretation of executable choreographies note that the execution environment is assumed to not only exchange the actual business messages carrying business content, but also some technical control messages that ensure compliance to the defined control flow. However, those technical control messages do not have business meaning so that their exchange can be auto-generated (cf. [5]).

At the beginning of figure 2's choreography, BTA 'act-id-1' immediately is enabled and starts upon request by the 'Customer' role. Upon acknowledgment by the 'Supplier' role, the BTA is performed according to a full-fledged

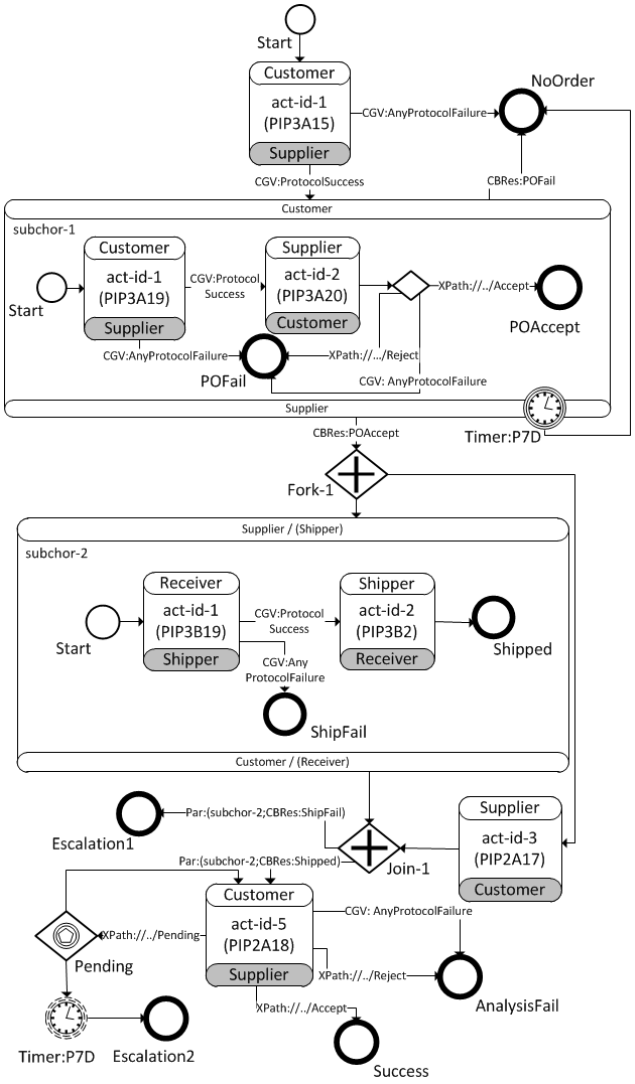

Figure 2. Sample Executable Choreography

execution protocol that produces a synchronized result between the two participating roles (cf. section II). The ebBP CGV language (cf. section II) is used to capture general protocol outcomes. For example, 'CGV:AnyProtocolFailure' is used to capture arbitrary protocol failures and the end state 'NoOrder' is reached upon that result. Note that even protocol failures can be used for routing because the execution protocol does not start before the 'Supplier' has acknowledged the request for the BTA so that both roles are aware of the protocol start. In case of a protocol success, the transition to 'subchor-1' is enabled. Beyond generic protocol outcomes, expressions on the exchanged business documents can be used for determining control flow and the standard ebBP expression languages are eligible for that. For example, some results of BTA 'act-id-5' at the bottom of figure 2 are captured using XPath expressions. In accordance with the BPMN standard, the expressions are underspeci-

fied (cf. [6, section 8.3.6]) and need to be extended upon implementation. Note that any content-based expression implicitly is AND-connected with 'CGV:ProtocolSuccess' because evaluating business contents only is sensible if the underlying exchange protocol has been successful.

Once the transition to BCA 'subchor-1' is enabled, it starts upon request by either the 'Customer' or the 'Supplier' and the semantics for performing choreographies is simply reapplied to it. The top-level choreography remains in state 'subchor-1' until the BCA has terminated. Note that the interrupting timer associated on the lower right of the sub-choreography shape is controlled by the BCA itself. "Interrupting" means that upon reaching the specified deadline the currently active state of the sub-choreography is terminated whereas a non-interrupting timer would result in waiting until the BCA has completed. In both cases, however, if a timeout is fired the subsequent state of the top-level choreography is determined by the outgoing transition of the timer shape. Otherwise, the follow-on state of the BCA is determined by the set of outgoing transitions of the BCA and the names of the end states of the BCA are used for routing (referred to as CollaborationResult (CBRes) expression language). So, if end state 'POFail' of BCA 'subchor-1' is reached then end state 'NoOrder' of the top-level choreography is reached and the according expression is 'CBRes:POFail'. If end state 'POAccept' of BCA 'subchor-1' is reached then the parallel structure demarcated by a fork node and join node is switched to.

Once the fork node of the parallel structure is reached, each of its branches is enabled. Basically, each branch is interpreted as one single BCA and therefore the semantics for performing choreographies can be reapplied which also includes starting a choreography upon request by one of the BCA's roles. If an explicit sub-choreography or call choreography shape is not used as depicted for the right-hand branch of the parallel structure in figure 2, then the first shape of the branch is interpreted as the follow-on state of a *virtual BC's* start state. The transition of that branch that terminates in the join node of the parallel structure is interpreted as ending up in an end state of the virtual BC. If the branch links to any additional end states then these are interpreted as additional end states of the virtual BC as well. While such a virtual sub-choreography concept is not desirable from a software-engineering point of view, it had to be introduced for [9] in order not to require users to specify an explicit sub-choreography or call choreography shape for each branch of a parallel structure. The join node of a parallel structure is reached once all branches have terminated. As fork and join nodes always must be paired, it is legal to interpret the whole structure as just another state of the top-level choreography. The "Par" language has been invented to capture the results of parallel structures once all branches have terminated. Expressions in the Par language simply are two-tuples consisting of the activity id of one the branches and a CBRes expression for capturing the result of the identified branch's result. For example, 'Par:(subchor-2;CBRes:ShipFail)' evaluates to true if the 'subchor-2' branch of the parallel structure terminates in end state 'ShipFail'.

Finally, the event-based choice 'Pending' exemplifies control flow determination by means of events. Once reached, all outgoing transitions of the event-based choice are activated. Although the 'Pending' state only is connected to BTA 'act-id-5' and a timer, an arbitrary number of BTAs or BCAs could be connected. The semantics is such that only one of the outgoing branches can be activated and the integration partners are assumed to use technical control messages for excluding the activation of multiple follow-on activities at the same time (an according mechanism is provided in [5]).

It is worth noting that the execution of such executable choreographies is almost completely covered in [5], in particular the interplay between business messages and technical control messages. Only the mechanism for timeout processing has to be imported from [4].

SeqMP is the second strict B2Bi choreography class that is visualized using BPMN. While executable choreographies target binary choreographies, SeqMP targets multi-party choreographies. Providing a format for multi-party choreographies is mandated by the sheer existence of the concept of supply chains, but the analysis of RIGs (cf. above) reveals that most systems are implemented on a bilateral basis. In consequence, SeqMP models a multi-party choreography as sequence of binary executable choreographies with changing roles and looks at analysis instead of implementation.

Figure 3 shows a sample SeqMP choreography that has been adapted from RosettaNet's "eBusiness process scenario library"[4] and depicts a 'Customer', 'Seller' and 'Shipper' role of an order to cash scenario. First the 'Customer' orders products at the 'Seller' (BCA 'c1') who then demands logistics services from the 'Shipper' (BCA 'c2'). The 'Shipper' delivers the products to the 'Customer' (BCA 'c3') who then pays the products (BCA 'c4'). Each BCA is assumed to be an executable choreography in order to ensure synchronized results among the interacting roles. As long as result synchronization is catered for, other styles for BCA modeling are conceivable. However, note that SeqMP choreographies are not eligible as components of other SeqMP choreographies because results are not synchronized among all roles.

The paradigm for SeqMP models is again a state machine with the start states, end states and BCAs as states. Parallel structures, event-based choices, and hierarchical decomposition beyond including executable BCAs is not admissible. The interpretation of SeqMP choreography modeling ele-

[4] http://www.rosettanet.org/Support/ImplementingRosettaNetStandards/eBusinessProcessScenarioLibrary/tabid/3319/Default.aspx

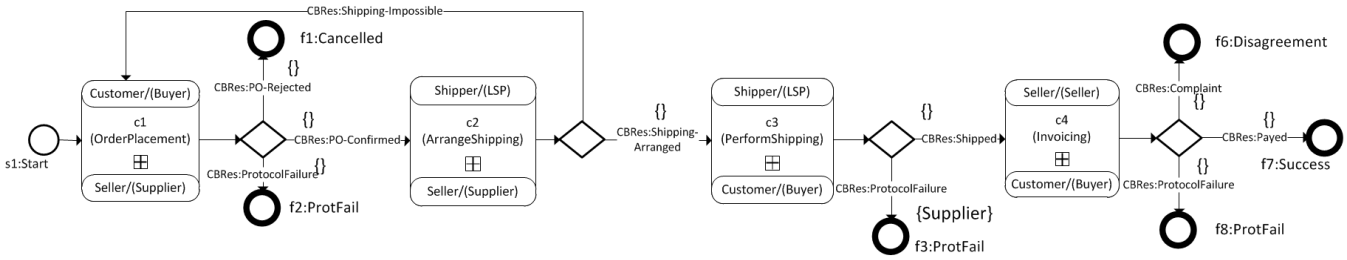

Figure 3. Sample SeqMP Choreography

ments is almost the same as for executable choreographies except for triggering BCAs and escalation sets.

For the SeqMP case, BCAs typically cannot be triggered by all participating roles. For example, BCA 'c1' is performed between the 'Customer' and 'Seller'. Therefore, the 'Shipper' neither knows about the result nor about the fact that the BCA has been performed at all. As a consequence, it is logical that only the 'Seller' can trigger BCA c2. Consistently, it is required that any two subsequent BCAs of SeqMP models must share at least one role.

Escalation sets are used for capturing synchronization deficits that may result from firing a particular transition. As an example, consider the transition to final state 'f3:ProtFail'. After BCA 'c2' has been performed successfully, the 'Supplier' role still expects to receive payment information from the 'Customer' in BCA 'c4'. However, if BCA 'c3' fails the overall choreography stops and the 'Seller' would need information about that. Practice teaches that this cannot be assumed to happen. So, firing the transition to 'f3:ProtFail' establishes a synchronization deficit of the 'Supplier' which is explicitly captured by including the 'Supplier' in that transition's escalation set. Note, that synchronization deficits not always are as obvious as in the sample of figure 3. Algorithms for the configurable analysis of synchronization deficits as well as for creating role projections are available in [8]. Note, however, that [8] does not come up with automatic resolutions of the synchronization deficits. Instead, it provides a framework for identifying synchronization deficits and leaves the resolution to partner agreements.

VI. BPMN Compliance and Validation

Validity of the proposed B2Bi choreography visualization calls for discussion regarding usability, BPMN compliance as well as completeness and soundness.

Section III motivated the approach of mapping the B2Bi choreography classes ebBP-ST [4], ebBP-Reg [5] and SeqMP [8] to BPMN. This significantly constrains the freedom for optimizing the visualization in terms of usability so that an empirical usability study has not been performed. However, the adoption by RosettaNet's MCC team that included experienced B2Bi experts of solution providers as well as B2Bi practitioners suggests that a reasonable level of usability is available.

BPMN compliance is identified in section III as subordinate goal to providing adequate B2Bi domain constructs and being able to provide the unambiguous semantics of ebBP-ST [4], ebBP-Reg [5] and SeqMP [8]. Therefore, the deviations from the BPMN standard described below deliberately are accepted. In that regard, note that restrictions and amendments to the BPMN choreography standard for the purpose of B2Bi choreography modeling are a natural thing because BPMN choreographies have not been designed as dedicated B2Bi choreography language.

The following BPMN extensions are defined:

1) Choreography tasks are interpreted as BTs that require a full-fledged execution protocol (cf. section II).
2) Expression languages for evaluating the result of BT executions are imported from ebBP (which is allowed by BPMN [6, section 10.3]).
3) Expression languages for capturing the result of BCAs and parallel structures are defined.
4) Basic rules for labeling top-level choreographies are defined (see [9] for details).
5) A notation for role mapping from choreography roles to subordinate choreography roles is defined.

The following BPMN rules are violated:

1) Guards are allowed to be added to transitions without a so-called "mini-diamond marker" which contradicts the following BPMN rule:
"A conditional outgoing Sequence Flow from an Activity MUST be drawn with a mini-diamond marker at the beginning of the connector" [6, section 8.3.13].
2) For an exclusive gateway, no evaluation order of conditions is prescribed which contradicts the following BPMN rule (completeness and disjointness are required instead, cf. [9]):
"In order to determine the outgoing Sequence Flows that receives the token, the conditions are evaluated in order" [6, section 13.3.2].
3) For sub-choreographies and call choreographies, more than one initiator band can be defined.
4) There is no distinction between collapsed (expanded) sub-choreographies and collapsed (expanded) call chore-

ographies by means of "line thickness" as in BPMN. Instead, only the shapes for collapsed call choreographies and expanded sub-choreographies are defined. There is only the concept of BCA that unifies the concepts of call choreographies and sub-choreographies. The use case of just defining part of a larger choreography is supported by implicit role mapping while the use case of using a BCA in several other choreographies is enabled by explicit role mapping. However there is no need to define two different types of BCA representations for that.

Therefore, the following visualization rules of BPMN are disregarded:

"If the Call Choreography calls a Choreography, then there are two options:

- The details of the called Choreography can be hidden and the shape will be the same as a collapsed Sub-Choreography, but the boundary of the shape MUST have a thick line (see Figure 11.25).

- The details of the called Choreography can be shown and the shape will be the same as an expanded Sub-Choreography, but the boundary of the shape MUST have a thick line (see Figure 11.26)" [6, section 11.4.3]

5) Conditional expressions after join nodes (parallel gateways) are allowed for in order to capture the result of a parallel structure. This contradicts the following rule: *"A source Gateway [of a conditional sequence flow] MUST NOT be of type Parallel or Event"* [6, section 8.3.13].

6) The follow-on BTs of an event-based choice may have different senders and receivers. For example there could be two BTs A and B where role1 is assigned the requester role of A and the responder role of B and role2 is assigned the responder role of A and requester role of B. This contradicts the following BPMN rule: *"On the right side of the [event-based] Gateway: either - the senders MUST to be the same; or - the receivers MUST to be the same."* [6, section 11.6.2]

In order to cater for completeness and soundness beyond mapping established B2Bi choreography classes, 11 use cases taken from the RosettaNet RIG library as well as the RosettaNet "Order to Cash eBusiness Scenarios" have been modeled using the above modeling guidelines. The use cases have been selected such that the different control flow constructs of the above guidelines are covered. As this use case analysis was part of the development of the above modeling guidelines, all use cases could be modeled and the result is available together with the corresponding ebBP specifications as part of the *RosettaNet Methodology for Choreography Modeling* deliverables[5]. Those ebBP models were generated by hand with the following two purposes. First, to find out whether or not ebBP *skeletons* can be derived automatically from the information available in the BPMN model and, second, to analyze the detail that has to be completed in order to create a complete ebBP specification from such a skeleton. The derivability of ebBP skeletons ensures the applicability of the semantics defined for ebBP-ST, ebBP-Reg and SeqMP as ebBP has been used as representation format for each of the different B2Bi choreography classes. The technical gap that needs to be filled in when specifying complete ebBP models mainly concerns the configuration of BTs, i.e., the business signals or business document versions to be used as well as reliability and security properties. How such models can be turned into BPEL-based implementations is described in [4], [5], [11].

[5]http://www.rosettanet.org/dnn_rose/DocumentLibrary/tabid/2979/DMXModule/624/Command/Core_Download/Method/attachment/Default.aspx?EntryId=9858

VII. Related Work

There is a considerable amount of work in the area of choreography modeling. For example, [15] show how to extend BPEL for choreography modeling and [16] have come up with a visual notation for choreography specification. However these approaches target at service choreographies and do not offer sufficient B2Bi domain concepts.

In the area of B2Bi choreography modeling, less research work is available. [17] propose the so-called Business Choreography Language (BCL) as visual notation, but a formal characterization of valid models is not given. As this language is not standardized, it has not been considered as visualization option for the work at hand.

BPMN and UMM both offer visual notations for choreography specification and both do not define valid models formally. Note that there is a difference between notation and grammar rules that ensure validity. Notations like BPMN or UMM that target at a large variety of scenarios hardly can come up with such grammar rules in order not to over-constrain the notation. Even the *RosettaNet Methodology for Creating Choreographies* [9] proposes, in addition to the choreography classes presented here, *cartography choreographies* that do not follow strict modeling guidelines in order to cater for unusual B2Bi scenarios.

Instead of BPMN, UMM could have been used as visual notation for the B2Bi choreography classes of this work. Indeed, UMM is a dedicated B2Bi notation and therefore outperforms BPMN in terms of offering domain concepts. However, it requires multiple views to be defined. BPMN then was chosen as visualization option in order to provide complete choreography specifications in one single diagram.

In [18], 'iBPMN' is proposed as adaptation of BPMN for choreography modeling. However, iBPMN is designed for a prior version of BPMN that did not include a section on interaction-style choreographies that are the basis for the visualization of the work at hand. Also, iBPMN is not specifically tailored to B2Bi and hence does not offer required domain concepts such as BTs.

VIII. Conclusion and Future Work

In this work, the BPMN choreography notation has been used for the purpose of B2Bi choreography modeling. The visual constructs used have been introduced and valid combinations of those constructs have been exemplified by means of sample processes. The visualization is aligned with established B2Bi choreography classes with formal characterizations of valid models and unambiguous semantics. These semantics can also be applied to the visual B2Bi choreographies proposed in this work.

The contribution of this work is not choosing BPMN as choreography notation, but rather restricting and adapting BPMN choreographies to a set of constructs and grammar rules that is adequate for B2Bi. However, this adaptation also implies giving up strict BPMN compliance. Hence, the definition of a dedicated B2Bi choreography profile of BPMN is desirable.

Future work targets at applying the approach of this work to UMM. The RosettaNet Methodology for Creating Choreographies explicitly allows for alternative visualizations as long as the modeling constructs of section IV can sensibly be represented (which is the case for UMM). Using UMM as visualization promises the amenability of this work to application areas that require multi-view modeling including explicit means for information modeling.

References

[1] A. Schönberger, "Do we need a refined choreography notion?" in *Proceedings of the 3rd Central-European Workshop on Services and their Composition (ZEUS), Karlsruhe, Germany, February 21-22, 2011*, ser. CEUR Workshop Proceedings. CEUR-WS.org, Feb 2011, pp. 16–23.

[2] H. J. Ahn, P. Childerhouse, G. Vossen, and H. Lee, "Rethinking XML-enabled agile supply chains," *International Journal of Information Management*, vol. article in press, 2011.

[3] OASIS, *ebXML Business Process Specification Schema Technical Specification*, 2nd ed., OASIS, December 2006.

[4] A. Schönberger, C. Pflügler, and G. Wirtz, "Translating shared state based ebXML BPSS models to WS-BPEL," *International Journal of Business Intelligence and Data Mining - Special Issue: 11th International Conference on Information Integration and Web-Based Applications and Services in December 2009*, vol. 5, no. 4, pp. 398 – 442, 2010.

[5] A. Schönberger and G. Wirtz, "Towards executing ebBP-Reg B2Bi choreographies," in *Proceedings of the 12th IEEE Conference on Commerce and Enterprise Computing (CEC'10), Shanghai, China*. IEEE, November 10-12 2010.

[6] OMG, *Business Process Model and Notation, v2.0*, OMG, January 2011. [Online]. Available: http://www.omg.org/spec/BPMN/2.0

[7] UN/CEFACT, *UML Profile for UN/CEFACT's Modeling Methodology (UMM) - Foundation Module*, 2nd ed., UN/CEFACT, 4 2011.

[8] A. Schönberger and G. Wirtz, "Configurable analysis of sequential multi-party choreographies," *forthcoming in International Journal of Computer Systems Science and Engineering*, March 2012.

[9] RosettaNet, *RosettaNet Methodology for Creating Choreographies, R11.00.00A*, RosettaNet, July 2011.

[10] ——, *Message Control and Choreography (MCC) - Profile-Web Services (WS), Release 11.00.00A*, RosettaNet, June 2010.

[11] A. Schönberger, G. Wirtz, C. Huemer, and M. Zapletal, "A composable, QoS-aware and web services-based execution model for ebXML BPSS BusinessTransactions," in *Proceedings of the 6th 2010 World Congress on Services (SERVICES2010), Miami, Florida, USA*. IEEE, July 2010, pp. 229 – 236.

[12] Y. Wand and R. Weber, "Research commentary: Information systems and conceptual modeling–a research agenda," *Info. Sys. Research*, vol. 13, no. 4, pp. 363–376, 2002.

[13] A. Schönberger, C. Wilms, and G. Wirtz, "A Requirements Analysis of Business-To-Business Integration," University of Bamberg, Technical Report: Bamberger Beiträge zur Wirtschaftsinformatik und Angewandten Informatik 83, 12 2009.

[14] H. A. Simon, *The sciences of the artificial*. The MIT Press, 1996.

[15] G. Decker, O. Kopp, F. Leymann, and M. Weske, "BPEL4Chor: Extending BPEL for modeling choreographies," in *Proceedings of the 2007 IEEE International Conference on Web Services (ICWS), July 9-13, 2007, Salt Lake City, Utah, USA*, 2007, pp. 296–303.

[16] J. M. Zaha, A. P. Barros, M. Dumas, and A. H. M. ter Hofstede, "Let's Dance: A language for service behavior modeling," in *Proceedings of the 14th international conference on cooperative information systems (CoopIS'06)*, Montpellier, France, 10 2006, pp. 145–162.

[17] M. Zapletal, T. Motal, and H. Werthner, "The business choreography language (BCL) - a domain-specific language for global choreographies," in *Proceedings of the 5th 2009 World Congress on Services (SERVICES 2009 PART II), Bangalore, India*. IEEE, September 2009.

[18] G. Decker and A. Barros, "Interaction modeling using BPMN," in *BPM'07: Proceedings of the 2007 international conference on Business process management*. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 208–219.

Acknowledgment

The author is deeply grateful and indebted to Hussam El-Leithy, Dale Moberg, Pankaj Telang, Christian Huemer, Marco Zapletal and all other members of the RosettaNet MCC team for fruitful discussions, feedback and inspirations on the topic of visualizing B2Bi choreographies.