# Using Variable Communication Technologies for Realizing Business Collaborations

Andreas Schönberger and Guido Wirtz
*Distributed and Mobile Systems Group*
*Otto-Friedrich-University of Bamberg*
*Bamberg, Germany*
$\{$*andreas.schoenberger* $|$ *guido.wirtz*$\}$*@uni-bamberg.de*

## Abstract

*In today's business world, enterprises are not only using Web Services for implementing B2Bi. Instead, AS2 for realizing Internet based EDI, ebXML Messaging and even SMTP and FTP are used for exchanging business documents. Harmonizing B2Bi communication technology in a way that solely uses Web Services is barely an option for large enterprises as this would result in losing investments in existing IT systems. Moreover, forcing a specific communication technology upon all business partners is neither intended nor realistic in practice. At the same time, existing business document exchanges frequently only implement B2Bi scenarios partly. This paper focuses on a method for composing existing and new business document exchanges without replacing communication technology already in use. First, ebXML BPSS is proposed for describing the choreography of business document exchanges in a technology agnostic way. Second, ebXML CPPA is used to define messaging characteristics at the level of ebBP BusinessTransactions. Third, an integration architecture for performing ebXML BPSS choreographies using BPEL processes that execute each integration partner's message exchanges is proposed. These BPEL processes assume Web service wrappers for reusing the functionality of messaging systems that are responsible for performing the actual business document exchanges and for providing sufficient status information to the caller. Different messaging systems then can be used for incorporating variable communication technologies in business collaborations.*

**Keywords:** B2Bi, micro-choreographies, ebXML BPSS, WS-BPEL

## 1. Introduction

Supply Chain Management (SCM) is a key success factor for enterprises in today's business world [1]. Hence, Business-To-Business Integration (B2Bi) as a core SCM task is an area of extraordinary importance and the challenging problems of this domain are to be thoroughly researched. First, personnel of integration partners with different background and vocabulary need suitable models as means for agreement. Second, integration partners have to implement IT systems that perform the necessary message exchanges at runtime and, at the same time, strictly conform to these models. Third, interoperable communication among typically heterogeneous IT systems of integration partners has to be achieved. To make the third aspect an even harder challenge, enterprises may also apply more than one communication technology for realizing B2Bi due to internal or external constraints. This may be the case because a single communication technology cannot be forced upon all integration partners or because of legal regulations, e.g., customs authorities may require enterprises to send customs declarations using AS2 [2]. Standardization organizations take this problem into account, too. RosettaNet[1], for example, as a leading international B2Bi consortium has released the "Multi Messaging Services" profiles[2] for describing how to exchange RosettaNet defined standard business documents via Web Services, AS2 and ebXML Messaging [3] in 2008. This situation amounts to the first research problem that is investigated in this paper:

**Problem 1:** *How can new complex business collaborations be built using more than one type of communication technology?*

In [4], we have introduced a top-down approach that tackles B2Bi challenges by first applying choreography models as means for agreement and then translating these choreography models into executable orchestration models in order to ensure conformance and to speed up development time. Thus, the overall message exchange of integration partners can be captured from a global point of view using choreographies while the individual sequence of message transmissions of each integration partner can be specified as orchestrations. ebXML BPSS [5] (ebBP in the following) has been chosen as choreography language due to its dedication to B2Bi and WS-BPEL [6] (BPEL in the following) has been proposed as orchestration language due to its position as de-facto standard in Web service based integration. Consequently, Web service technologies have been proposed for solving interoperability issues.

Yet, as a top-down approach, the work of [4] is not directly amenable to B2Bi scenarios that are to reuse existing

1. http://www.rosettanet.org, last accessed 15-06-2009
2. http://rosettanet.org/cms/sites/RosettaNet/Standards/RStandards/index.html, last accessed 15-06-2009

business document exchanges. As numerous B2Bi projects already have been realized today, new B2Bi projects are likely to be confronted with building complex business collaborations that reuse existing functionality. This situation gives rise to the question how existing business document exchanges and non Web service communication technologies can be incorporated in the process of translating ebBP choreographies into BPEL orchestrations for B2Bi as described in [4]. As this may be necessary for the same reasons as research problem 1 and, moreover, is indispensible for not losing investments spent on existing IT systems, this question leads to a second research problem that is as important as the first one:

**Problem 2:** *How can complex business collaborations be built from existing collaborations no matter which communication technology has been applied for implementing these existing collaborations?*

We approach both problems by investigating necessary design decisions, models and integration architecture on the abstraction levels of choreography and orchestration of a B2Bi as identified in [7]. The modeling of B2Bi collaborations on the business process level or even more abstract levels as well as the details of implementing private business logic are out of the focus of this paper. The rest of the paper is structured as follows: Section 2 describes preliminaries and section 3 discusses related work. Section 4 investigates the modeling concepts to be included on the integration levels identified. Then, section 5 shows how ebBP and ebXML CPPA [8], [9] (CPPA in the following) can be used to build according models for agreement. Section 6 describes an integration architecture that can be used to perform message exchanges at runtime as defined in the ebBP and CPPA agreements. Section 7 concludes the paper and points out directions for future work.

## 2. Basics

For the work at hand, basic knowledge about choreographies and orchestrations is assumed. Concerning technologies, ebBP, CPPA and BPEL are core to our approach.

The ebBP choreography language is based on the concept of *BusinessTransactions* (BT) that are used for exchanging one or two business documents between the so-called *RequestingRole* and *RespondingRole*. For the exchange of each business document, so-called *BusinessActivities* (BA) are used to specify whether *ReceiptAcknowledgements* and *AcceptanceAcknowledgements* as well as according exceptions are needed. So-called *BusinessCollaborations* with at least two roles (integration partners) can be combined to build complex integration models. Usual control flow constructs like *Decision*, *Fork* or *Join* can be used to choreograph *BusinessTransactionActivitys* (BTA) and *BusinessCollaborationActivitys* (CA) that specify the actual execution of *BusinessTransactions* and *BusinessCollaborations*, respectively, by adding execution parameters such as *TimeToPerform* and mapping the roles of the performing *BusinessCollaboration* to the roles of the performed entity.

CPPA is a standard that essentially serves for describing the technical communication capabilities of an integration partner in a collaboration protocol profile document (CPP) and for agreeing which of these to use when performing a *BusinessCollaboration* in a collaboration protocol agreement document (CPA). For doing so, the exchange of business documents and accompanying acknowledgements as defined in an ebBP choreography is associated with so-called **DeliveryChannels** using **ServiceBindings**. A **DeliveryChannel** defines which transport and messaging protocol characteristics to use for performing such an exchange by referencing **Transport** and **DocExchange** elements. Note that the same specification elements are used for creating CPP and CPA documents. For details please refer to [8], [9].

BPEL is a Web service orchestration language that is used for defining executable (or abstract) processes composed by a series of incoming or outgoing Web service calls. So-called `partnerLinkTypes` are defined within corresponding WSDL files that define `roles` in Web service communications based on WSDL `portTypes`. The BPEL process under consideration then uses `partnerLink` definitions for incorporating the `roles` of `partnerLinkTypes` and thus defines the functionality consumed or offered by the process. Based on these `partnerLinks` synchronous and asynchronous interactions like `invoke`, `receive` or `onMessage` can be defined and constructs like `sequence`, `if` and `while` can be used to define the control flow between these interactions. For details please refer to [6].

## 3. Related Work

Key aspects that can't be ignored when discussing interoperability matters have been presented in [10] that derives interoperability issues from existing business process definition languages and in [11] that describes system aspects concerned with interoperability issues within an *"interoperability framework for enterprise applications"*. However, these approaches do not specifically target B2Bi but can be used for contrasting with our results of section 4.

Using variable communication technologies for B2Bi is not a new idea as can be seen from the functionality of CPPA. In the work at hand we extend this idea to opaque BTs (cf. section 5) that hide technology details. Moreover, we describe the orchestration of such BT implementations at runtime using modular process structures. [12] propose using modular BPEL structures for implementing B2Bi but do not specify support for variable communication structures. In [13], a review of modularity in process models is performed. Not being targeted at runtime aspects, variable communication technologies have not been identified as relevant.

The idea of considering the encapsulation of business logic using backend interfaces and the separation of control flow in *control processes* is not new as well. Basically, the isolation of control flow is due to workflow research and dates back to the 1990s, e.g., [14]. In the B2Bi domain, RosettaNet applies the idea of separating *public* and *private* processes in its implementation framework [15]. Moreover, RosettaNet applies variable communication technologies to exchanging single business documents (cf. 1). We extend this idea to the composition of multiple business document exchanges using ebBP BTs as decomposition unit and applying control/master processes for encapsulating control flow logic.
We demonstrate how ebBP can be used to represent choreography models in a textual way. Other approaches like [16] provide a visual interface for modeling business collaborations and therefore obviously are much more user friendly when it comes to modeling and understanding complex choreography models. We do not consider ebBP to be an alternative to visually modeling business collaborations but rather as a common interchange format that may be derived from various visual languages and seems to be more suitable for further handling by analysis, transformation and execution machinery.

## 4. Modeling Requirements

This paper focuses on the interaction of B2Bi collaboration partners and particularly on the abstraction levels of choreography and orchestration. Clearly, the discussion of models for these abstraction levels requires the analysis of the concepts to be captured. We have performed an according analysis for the choreography layer (cf. [17], [4]) which leads to the following concepts:
**Business Documents:** From the early days of B2Bi until now, B2Bi is based on exchanging business documents such as quotes, orders and invoices. Numerous efforts for defining and exchanging business documents like EDIFACT[3], RosettaNet or UBL[4] empirically support this argument. Hence, business documents need adequate modeling.
**Shared states:** B2Bi is state driven as any business collaboration essentially changes states in the integration partners' IT systems by transmitting business documents. Obviously, such state changes, e.g., switching from *no quote available* to *quote available*, need to be synchronized among participants and therefore states can be considered to be concertedly left and reached by integration partners which is why we denote these as *shared* states. Considering the importance of state in the B2Bi domain, a suitable representation of shared state is indispensable.
**Micro-choreographies:** Exchanging and processing business documents is error prone. Technical communication errors as well as invalid content of business documents may prohibit performing business collaborations as intended. Consequently, the exchange of business documents is typically accompanied by some kind of acknowledgement messages. In order to provide an adequate abstraction, the sequence as well as the semantics of processing these messages shall be defined as protocols. Hence, we propose the notion of micro-choreographies for exchanging several business documents according to such protocols in an *all-or-nothing* fashion. It's the task of micro-choreographies to ensure consistent shared state changes of business collaborations.
**Control flow:** Complex business collaborations apparently are built from more than one shared state and micro-choreography. Therefore, control flow elements for composing shared states and micro-choreographies in sequences, loops and parallel structures are needed.
**Roles:** Roles are commonly accepted as a means to specify the tasks a collaboration participant is obliged to fulfill in an abstract yet focussed way. As such, modeling of roles should be supported.
**Quality of Service (QoS):** QoS frequently are defined as measurable non-functional qualities of network performance. In the B2Bi domain, special attention also has to be paid to non-functional qualities like encryption or authentication which are hard to measure. Therefore, QoS have to be specified in a way that matches B2Bi needs.
**Interface definition:** Contrasting the set of concepts identified above with related work [10], [11] shows that we do not directly consider system interfaces. This is due to the fact that interfaces of systems do not necessarily have to be modeled at the choreography layer as choreographies rather describe the *what* and not the *how* of exchanging business messages. But as the work at hand investigates both, the choreography and the orchestration layer, the requirement of representing system interfaces is adopted here.

## 5. Choreography Models for Agreement

ebBP is a choreography language that explicitly targets B2Bi and offers natural support for most of the modeling concepts identified in section 4. In particular, the import of business document definitions, the specification of control flow and roles as well as the specification of QoS properties relevant to B2Bi are directly supported. Moreover, the concepts of BT and *BusinessCollaboration* offer the possibility to represent micro-choreographies if the ebBP protocol for computing the status of a BTA (cf. [5] sec. 3.6.3) is enhanced with reliable messaging for message exchanges. As micro-choreographies transform a shared state into another shared state we propose to add transitions from a shared state to all micro-choreographies that are applicable in the respective state. Unfortunately, ebBP does not offer any modeling elements that correspond to shared states. In order to provide

3. http://www.unece.org/trade/untdid/welcome.htm
4. http://www.oasis-open.org/committees/ubl/

an equivalent control flow each transition that would link to a shared state then would have to be duplicated and linked to each follow-on micro-choreography of a shared state. Clearly, this might lead to problems in case a micro-choreography is linked to from multiple shared states. If so, the ebBP concepts of BTA and CA come in handy as they allow for reuse of BTs and *BusinessCollaborations* and thus are suitable to emulate shared states by creating a new BTA/CA each time a BT/*BusinessCollaboration* logically follows a different shared state. Finally, *interface definition* is partly supported in ebBP by means of *OperationMappings* that map business document and accompanying acknowledgement exchanges to abstract operations that can later on be dereferenced by WSDL operations (cf. [5] sec. 3.4.9.8).
Before discussing the details of modeling, the granularity of process modules that different communication technologies can be applied to has to be defined. As pointed out in section 4, the exchange of business documents within micro-choreographies shall be performed according to protocols for ensuring consistency. These protocols may make use of qualities provided by the communication layer such as reliability or addressing. Hence, allowing for multiple communication technologies within a single micro-choreography, e.g., using AS2 for sending a business document and Web Services for sending the corresponding acknowledgement, may increase the complexity of these protocols dramatically. Therefore, we only allow for a single communication technology for performing BTs as micro-choreography representations.
Having defined the granularity for applying different communication technologies, the specification of these must be decided upon. ebBP does offer abstract choreography models and, as such, does not provide direct support for this purpose. In consequence, ebBP either needs an extension or missing functionality needs to be provided by different technologies. Regarding research problem 1 identified in section 1, CPPA already offers functionality for specifying which communication technology to use for performing a BTA. CPPA offers the concept of **DeliveryChannels** for specifying which transport protocol and messaging protocol characteristics to use for exchanging business documents and acknowledgements. In the currently valid version 2.0 [8], CPPA offers HTTP, FTP and SMTP as transport protocols and ebXML Messaging (ebMS in the following) as messaging protocol. The draft of CPPA 3.0 further leverages Web Services and AS2 as messaging protocols. The CPPA **Transport** and **DocExchange** elements can then be used to provide detailed configurations for transport and messaging protocols.
Regarding research problem 2 of section 1, we propose not to use regular BTs for representing micro-choreographies because existing business document exchanges shall not be redefined using ebBP and CPPA functionality but reused instead. Therefore, the ebBP *DataExchange* type of business transaction should be used for defining an *opaque* micro-choreography, i.e., without business document and message exchange definitions. CPPA functionality should then be used to define how to trigger such an opaque BT.

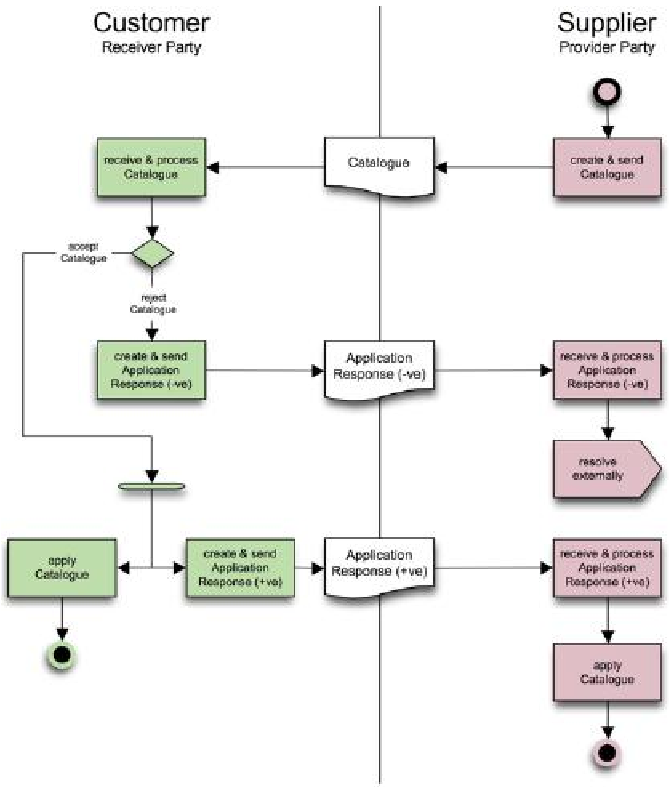
Customer
Receiver Party
Supplier
Provider Party
receive & process Catalogue
Catalogue
create & send Catalogue
accept Catalogue
reject Catalogue
create & send Application Response (-ve)
Application Response (-ve)
receive & process Application Response (-ve)
resolve externally
apply Catalogue
create & send Application Response (+ve)
Application Response (+ve)
receive & process Application Response (+ve)
apply Catalogue

Figure 1. Activity diagram of NES profile 1[18]

In the following, profile 1 of the Northern European Subset (NES) standard business processes [18] is used to demonstrate the above concepts. Figure 1 shows how the *supplier* role of NES profile 1 sends a catalogue document to the *customer* role which then accepts or rejects the catalogue and subsequently sends back a positive or negative application response to the *supplier*. In this scenario, sending a catalogue will be modeled as an opaque BT (research problem 1) and sending an application response as a new BT that is to be supported via variable communication technologies (research problem 2).
Listings 1 and 2 show the ebBP BT definitions the integration partners of NES profile1 have to agree upon. The opaque BT in listing 1 essentially only declares a BT and defines transaction scope roles that can be associated with collaboration roles later on. All other components of a BT such as the definition of business documents or acknowledgements are not specified because the transaction is opaque.

Listing 1. Opaque ebBP BusinessTransaction

```
<DataExchange name="distributeCatalogue"
  nameID="bt_distCat">
<!-- Roles needed for mapping Supplier/
     Customer -->
  <RequestingRole name="initiator"
  nameID="bt_distCat_initiator">
  </RequestingRole>
  <RespondingRole name="responder"
  nameID="bt_distCat_responder">
  </RespondingRole>
</DataExchange>
```

Opposed to that, listing 2 shows a *usual* ebBP BT that also defines the business document to exchange as well as accompanying acknowledgements and ebBP QoS attributes like *isAuthorizationRequired* or *isAuthenticated* that can be specified to the sending process or to the document itself. Note that the definition in listing 2 would not be different if only a single communication technology was allowed for.

Listing 2. Usual ebBP BusinessTransaction

```
<InformationDistribution
  name="distributeResponse"
 nameID="bt_distResp"
 isGuaranteedDeliveryRequired="true">
<!-- Requesting/RespondingRole similar to
    opaque transaction-->
 <RequestingBusinessActivity
   name="send app response"
  nameID="bt_distResp_ba_req"
  isAuthorizationRequired="true"
  ... more QoS attributes>
  <DocumentEnvelope name="app response"
   businessDocumentRef="bd_appResponse"
   nameID="bt_distResp_doc_appResponse"
   isAuthenticated="transient"
   ... more QoS attributes>
  </DocumentEnvelope>
  <!-- ReceiptAcknowledgement/
  ReceiptAcknowledgementException
  left out for brevity!
  -->
 </RequestingBusinessActivity>
</InformationDistribution>
```

While the usage of *usual* BTs within *BusinessCollaborations* also is no different whether variable communication technologies are allowed for or not, opaque transactions again need special treatment. Listing 3 shows the execution of the BT from listing 1 and the follow-on routing of the collaboration using an ebBP *Decision*. At first sight, this definition looks very much the same as for *usual* BTs. But normally, ebBP *ConditionExpressions* within *Decisions* may reference business document instances, e.g., an XPATH expression may capture the value of some business document attribute. As no business documents are defined within an opaque transaction such expressions are not valid. Therefore we propose to use ebBP *ConditionGuardValues* as used in listing 3 and a new expression language *OpaqueMappingValue* which is allowed for according to [5], section 3.4.11.1.1. ebBP *ConditionGuardValues* specify generic types of processing results of a transaction like *ProtocolSuccess* or *AnyProtocolFailure* which can generically be mapped to opaque transactions. *OpaqueMappingValues* can be used to denote a set of different business outcomes represented by strings which then would have to be computed by the system that performs the transaction at runtime.

Listing 3. Opaque ebBP BTA with Decision

```
<BusinessTransactionActivity
 businessTransactionRef="bt_distCat"
 name="send catalogue"
 nameID="bta_sendCat">
 <TimeToPerform duration="PT1H"
  type="design"/>
 <Performs
  currentRoleRef="profile1global_supplier"
  performsRoleRef="bt_distCat_initiator"/>
<!-- analog Performs tag for customer -->
</BusinessTransactionActivity>

<Decision
 name="after send catalogue"
 nameID="dec_sendCatalogue">
 <FromLink
  fromBusinessStateRef="bta_sendCat"/>
 <ToLink
   toBusinessStateRef="techFail">
  <ConditionExpression
   expressionLanguage="ConditionGuardValue"
   expression="AnyProtocolFailure"/>
 </ToLink>
 <!-- More ToLinks -->
</Decision>
```

While the ebBP examples above are the same for both integration partners, the following CPPA examples are specific for the supplier role of NES profile 1. Nonetheless, these definitions are agreed upon by both integration partners. Listings 4 and 5 show the definition of transport and messaging protocol characteristics for the BT of listing 2. In this transaction, the supplier uses HTTP as transport protocol and ebMS as messaging protocol. Accordingly, listing 4 defines concepts like an **Endpoint** for accepting HTTP connections and listing 5 defines concepts like the number of retries for message exchanges. Note that different **Transport** and **DocExchange** elements could be defined for the same BT which enables using variable communication technologies (research problem 1).

Listing 4. CPPA Transport Definition

```
<Transport transportId="transportEbMSSup">
 <!-- TransportSender left out -->
 <TransportReceiver>
  <TransportProtocol version="1.1">
   HTTP</TransportProtocol>
  <AccessAuthentication>digest</A..n>
  <Endpoint
   uri="http://www.supplier.com/ebMSHandler
       /distCatResp"/>
  <TransportServerSecurity>
   <TransportSecurityProtocol
    version="1.2">TLS</T..l>
    <!-- more subtags -->
   </TransportServerSecurity>
 </TransportReceiver>
</Transport>
```

Listing 5. CPPA ebMS DocExchange

```
<DocExchange docExchangeId="dE_EbMSSup">
  <ebXMLReceiverBinding version="2.0">
    <ReliableMessaging>
      <Retries>3</Retries>
   <!-- more subtags -->
```

```
    </ReliableMessaging>
    <PersistDuration>P1D</PersistDuration>
    <ReceiverNonRepudiation>
      <NonRepudiationProtocol>http://www.w3
          .org/2000/09/xmldsig#</N..l>
   <!-- more subtags -->
    </ReceiverNonRepudiation>
    <ReceiverDigitalEnvelope>
      <DigitalEnvelopeProtocol
       version="2.0">S/MIME</D..l>
   <!-- more subtags -->
    </ReceiverDigitalEnvelope>
  </ebXMLReceiverBinding>
</DocExchange>
```

Finally, listing 6 shows a proposal for a new document exchange binding (cf. [9], section 4.3.10) for opaque BTs. This binding is fundamentally different from the binding in listing 5 as it defines identification information for the messaging system that is responsible for executing an opaque transaction instead of defining how the actual business document exchange has to be performed. At runtime, the supplier would have to map the information contained in the `<Exchange>` element to the local messaging system that controls the actual execution of an opaque transaction.

Listing 6. CPPA opaque DocExchange binding

```
<DocExchange docExchangeId="dE_OpaqueSup">
 <opaqueExchangeSenderBinding
  version="1.0">
  <!-- Identify opaque Transmission -->
  <Exchange>
   <B2BiLink>SupplierToCustomer</B2BiLink>
   <ExchangeIdentifier>
    CatalogCoordination-distCat-v3</E..r>
  </Exchange>
 </opaqueExchangeSenderBinding>
</DocExchange>
```

At the beginning of this section, we proposed to use BTs or *BusinessCollaborations* to represent micro-choreographies. In consequence, using variable communication technologies for micro-choreographies also calls for a solution for *BusinessCollaborations*. In the approach described above existing functionality (research problem 2) is incorporated as opaque transactions. As the structure of these transactions is unknown, trying to apply decomposition to opaque transactions does not make sense. Therefore, *BusinessCollaborations* are only used for composing BTs which then may use different communication technologies.

## 6. Integration Architecture

This section describes how BTs and *BusinessCollaborations* composed from such BTs can be executed using orchestration models. Note that, by mapping the concepts of section 5, we also address the required modeling concepts as defined in section 4. First, the realization of *usual* BTs is discussed. We propose to encapsulate business logic using backend interfaces and to use dedicated *control processes* for specifying the exact flow of message exchanges. Backend interfaces are in charge for creating and evaluating business documents, e.g., *accepting a catalogue by creating an according response*, and capturing real-world events, e.g., *a new catalogue has to be distributed*. Control processes have the task of interacting with partner processes on the one hand and backends on the other hand by accepting and forwarding messages as defined by the ebBP protocol for performing BTAs. The resulting flow of messages between backends and control processes that results from performing the BT defined in listing 2 is depicted in an idealized form in figure 2. There, the customer first captures the

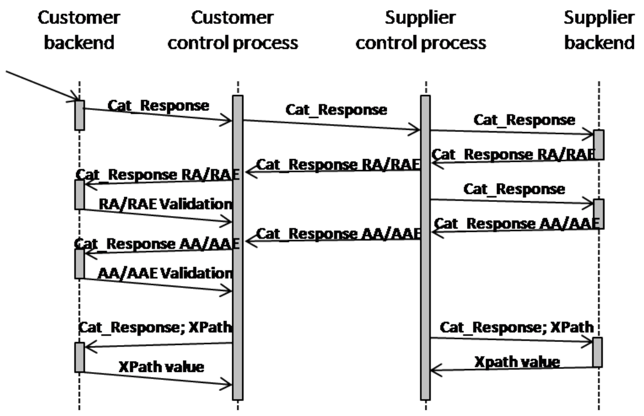

Figure 2. Message flow of NES profile 1 response

event that the response document for a recently received catalogue has been created and then uses her backend to send the response document (denoted Cat_Response) to the local control process. The customer's control process then forwards the response document to the supplier's control process which in turn forwards the business document to the supplier's backend. The document then gets stored, checked for readability and an analysis whether the document can be accepted for business processing is performed. Accordingly, the backend creates ebBP *Receipt/AcceptanceAcknowledgements* (RA/AA) or the corresponding exceptions (RAE/AAE). The supplier's control process is responsible for ensuring that readability and acceptance analysis can be performed in parallel and that timeout values as defined in ebBP are controlled. The supplier's backend acknowledgements or exceptions are forwarded by the supplier's control process to the customer's control process and then are sent for a validity check to the customer's backend. At the end of all these message exchanges and in case no processing errors occurred the business result of the BT is evaluated. In ebBP, XPath and other languages may be used to define expressions based on business documents. At runtime, these expression values can then be used for taking routing decisions. Accordingly, the control processes send the latest business document together with an expression to the backend systems which subsequently return the result value. Basically, various communication technologies could be applied to realize the message flow as defined above. In

[4], the translation of ebBP models into BPEL processes for implementing control processes and encapsulating backends as Web Services is described. Using ebBP and CPPA information as described in section 5, similar approaches for ebMS and AS2 are conceivable.
The realization of opaque BTs is trivial as these already are implemented.
Finally, we propose to implement *BusinessCollaborations* by applying the concept of separation between control processes and backend processes at a higher abstraction level. For the sake of clarity, control processes at this higher level are denoted *master* processes. These master processes essentially do not exchange business documents at all. Instead they let the implementations of BTs perform the document exchanges. For doing so, we propose to define wrappers for the local BT implementation parts of integration partners. These wrappers can then be used by the master processes to trigger the execution of a BT and get notified about the result upon termination of the BTA. Building wrappers equally applies to the realization of new *usual* BTs (research problem 1) as well as opaque BTs (research problem 2). Figure 3 exemplifies the interaction between master processes and lower level BTA implementations using NES profile 1. In this scenario, the supplier captures the event that a new instance of NES profile 1 has to be performed and uses her backend to send an initialization message to the supplier's master process. The supplier's and the customer's master processes then may exchange several control messages for exchanging an instance id or for negotiating timeouts as defined in the ebBP *BusinessCollaboration*. The details of control message exchanges are not shown. Subsequently, the supplier's master process notifies the supplier's backend via the corresponding wrapper that the BT for distributing the catalog can be performed which is defined as opaque (therefore, the black bars are included in the figure). At this point in time, control is handed over to the messaging system performing the opaque transaction. This transaction is then performed without participation of the master processes using the endpoints that are well-known to the system (as it already existed before) using, e.g., AS2. At the end of the opaque transaction the result is deduced and then notified to the supplier's master process. According to this result the supplier's master process may cancel the overall process or notify the customer's master process that the second BT of NES profile 1 can be performed. The customer's master process then uses the wrapper for the second BT, i.e., for exchanging the catalogue response document. Again, the transaction is performed without participation of the master processes. Therefore, there is no difference between opaque and non-opaque BTs from the point of view of interaction with master processes. Finally, the result of the catalogue response transaction is notified to the customer's master process and the collaboration gets terminated. ebBP CAs then conveniently can be mapped by providing wrappers for accessing master processes. Looking at possible im-

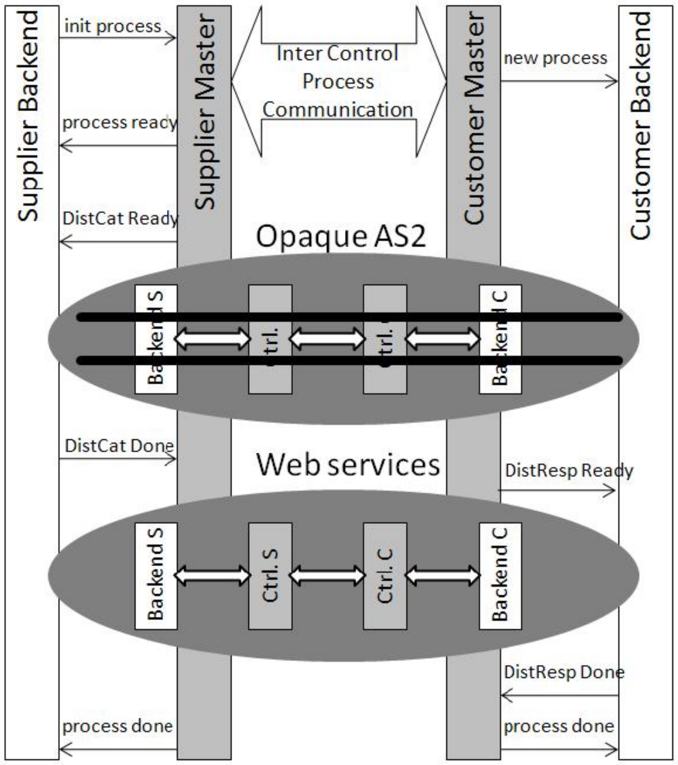

Figure 3. Master process/ BTA protocol interaction

plementation technologies, we propose BPEL for realizing master processes and Web services for encapsulating BT implementations. While other technologies are conceivable, BPEL and Web services clearly have advantages in terms of interoperability. This is relevant for interaction between master processes as well as between master processes and local BT implementations. Following this approach does not support research problem 1 in full because the realization of *BusinessCollaborations* then is tied to Web service technology. Indeed, other interface technologies like CORBA might be used but current research rather does not follow this direction. Note, that alternative *BusinessCollaboration* implementations may still be incorporated in our BPEL based approach by defining an opaque BT that represents the collaboration.
The approach of separating master processes and transaction processes imposes some constraints on the BT implementations that are described in the following:
***Availability of Web service wrappers:*** BPEL-based master processes must be able to access the local BT implementations which requires that these implementations can be wrapped by a Web service. From a technical perspective, this should not be too big a problem because Web services are a dedicated wrapping technology. From a business perspective, commercial off-the-shelf software might be used for BT implementations and interaction partners therefore may not have the necessary tool set for automatically triggering

transactions. In this case, a work flow based implementation of the Web service wrapper that requires human involvement might be a solution. Moreover, the wrapper of a BT implementation has to offer a Web service endpoint. We propose to use a separate endpoint for each BT type so that technology specific details of separating BT types do not have to be cared for on the master level.

***Start/Stop events and computable result:*** As master processes control when a BT is to be performed, a BT implementation wrapper has to offer a start and a stop event. Moreover, master processes are responsible for implementing the control flow between the BTs of a *BusinessCollaboration*. Therefore, the result of a BT execution must be computable, i.e., whether the execution succeeded from a protocol perspective and which of multiple business result values has been achieved. We argue that the application of all-or-nothing semantics to BT implementations helps in computing unambiguous result values.

***Process-Awareness:*** The BTs of a *BusinessCollaboration* are semantically related, e.g., the catalogue response document of NES profile 1 shall be related to the catalogue distributed before. Therefore, a BT implementation wrapper must be capable of mapping an instance id received from a master process to business content like quotes, orders or invoices. A (partner local) central storage containing such mapping data that is accessible from all BT implementation wrappers can provide the necessary infrastructure for doing so.

## 7. Conclusion and Future Work

In this paper, we described how to model the application of variable communication technologies to business collaborations on a choreography layer and we proposed an architecture for executing such models. ebBP and CPPA offer the tool set for describing how to implement variable communication technologies on a BT level and BPEL can be used for orchestrating these BT implementations at runtime. Future work includes implementing real world use cases according to the architecture for proving the feasibility in practice. Further, the results of [4] are promising that master processes and WSDL interfaces for encapsulating BT implementations can be generated from ebBP and CPPA models.

## References

[1] D. M. Lambert and M. C. Cooper, "Issues in supply chain management," *Industrial Marketing Management*, vol. 29, no. 1, pp. 65 – 83, 2000.

[2] D. Moberg and R. Drummond, *RFC 4130: MIME-Based Secure Peer-to-Peer Business Data Interchange Using HTTP, Applicability Statement 2 (AS2)*, The Internet Engineering Task Force (IETF), jul 2005.

[3] OASIS, *ebXML Messaging Services Version 3.0: Part 1, Core Features*, OASIS, October 2007.

[4] A. Schönberger, T. Benker, S. Fritzemeier, M. Geiger, S. Harrer, T. Kessner, J. Schwalb, and G. Wirtz, "QoS-enabled business-to-business integration using ebBP to WS-BPEL translations," in *Proc. of the IEEE SCC 2009 Int. Conf. on Services Computing*. IEEE, September 2009.

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

[6] ——, *Web Services Business Process Execution Language*, 2nd ed., April 2007.

[7] A. Schönberger and G. Wirtz, "Taxonomy on consistency requirements in the business process integration context," in *Proceedings of 2008 Conf. on Software Engineering and Knowledge Engineering (SEKE'2008)*. Redwood City, California, USA: Knowledge Systems Institute, 1.-3. July 2008.

[8] OASIS, *Collaboration-Protocol Profile and Agreement Specification Version 2.0*, OASIS, sep 2002.

[9] ——, *ebXML Collaboration-Protocol Profile and Agreement Specification Version 3.0 (Editors Draft, latest state as of June 2009)*, OASIS. [Online]. Available: http://www.oasis-open.org/committees/download.php/31996/ebcppa-v3.0-Spec-wd-r01-en-pete4.pdf

[10] L. B. Merino and G. B. Elguezabal, "Business process definition languages versus traditional methods towards interoperability," in *Proc. 4th Int. Conf. of COTS-Based Software Systems, ICCBSS 2005, Bilbao, Spain, February 2005*, 2005.

[11] B. Elvester, A. Hahn, A.-J. Berre, and T. Neple, "Towards an interoperability framework for model-driven development of software systems," in *Proceedings of the 1st Int. Conf. on Interoperability of Enterprise Software and Applications*, Switzerland, 2005, pp. 409–420.

[12] J.-H. Kim and C. Huemer, "From an ebXML BPSS choreography to a BPEL-based implementation," *SIGecom Exch.*, vol. 5, no. 2, pp. 1–11, 2004.

[13] H. Reijers and J. Mendling, "Modularity in process models: Review and effects," in *BPM '08: Proceedings of the 6th International Conference on Business Process Management*. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 20–35.

[14] F. Leymann and D. Roller, "Workflow-based applications," *IBM Syst. J.*, vol. 36, no. 1, pp. 102–123, 1997.

[15] *RosettaNet Implementation Framework: Core Specification*, V02.00.01 ed., RosettaNet, www.rosettanet.org, March 2002.

[16] Christian Huemer et al., *UN/CEFACTs Modeling Methodology (UMM): UMM Meta Model Foundation Module Version 1.0*, 1st ed., UN/CEFACT, 10 2006.

[17] A. Schönberger and G. Wirtz, "Using Webservice Choreography and Orchestration Perspectives to Model and Evaluate B2B Interactions," in *The 2006 Int. Conf. on Software Engineering Research and Practice (SERP'06)*, June 2006.

[18] *Northern European Subset Profiles*, 2nd ed., Northern European working group for development of a subset for UBL 2.0, July 2007. [Online]. Available: http://www.nesubl.eu