Association for Information Systems
AIS Electronic Library (AISeL)

PACIS 2010 Proceedings

Pacific Asia Conference on Information Systems (PACIS)

2010

A Comparison of Visual Modeling Notations for Web Services Choreography

W. L. Yeung
Lingnan University, wlyeung@ln.edu.hk

A COMPARISON OF VISUAL MODELING NOTATIONS FOR WEB SERVICES CHOREOGRAPHY

W. L. Yeung, Department of Computing and Decision Sciences, Lingnan University, Hong Kong, wlyeung@ln.edu.hk

Abstract

The Web Services Choreography Description Language (WS-CDL) is an XML-based language for specifying business protocols for \emph{web services enabled} collaborative processes. The use of visual notations in modeling web services choreography has so far been done in an ad hoc fashion as seen in the literature. This paper presents a choreography example in four different visual modeling notations and compares them with regard to the semantics of WS-CDL. The results are useful for establishing a reliable visual approach to modeling web services choreography.

Keywords: visual modeling, web services, choreography.

1 INTRODUCTION

Web services have emerged during the past decade as a prominent technology for supporting enterprise application integration (EAI) and business process management (BPM) within an organization (Lim & Wen, 2003; Zhao & Cheng, 2005; Albrecht, Dean, & Hansen, 2005). Applications from across different geographic and functional units of the same enterprise running on diverse platforms and infrastructures can be implemented or packaged as web services that are *orchestrated* to run according to visually defined workflow models and monitored by sophisticated business activity monitoring (BAM) tools.

The service oriented architecture (SOA) also enables the integration of BPM with B2B collaboration (Kim & Segev, 2005; Chen, Zhang, & Zhou, 2007). Enterprise applications implemented or packaged as web services are loosely coupled and can be dynamically bound together during the execution of a business process. This is most relevant to B2B collaborative processes in which business partnerships can be dynamically set up with interoperable web services running on diverse platforms, communicating and collaborating through the Internet.

SOA involves a stack of XML-based standards for enabling secure, reliable, self-describing interoperable Web services, as well as for modeling and implementing workflow or process-oriented web applications. The latter kind of standards includes the Web Service Business Process Execution language (WS-BPEL) for orchestrating a set of web services in a business process. While web services for a multi-party collaborative process can in principle be orchestrated by a dominant or independent party according to an overall cross-organizational workflow model, businesses are not necessarily willing to delegate control of their internal processes or reveal them to other parties (Kavantzas et al., 2005). Instead a *web services choreography* can be specified to serve as a contract containing a *global* definition of the common ordering conditions and constraints under which message are exchanged among the participants in a collaborative process (Kavantzas et al., 2005). Participants must agree on such a choreography and orchestrate their respective web services in conformance with it. While orchestration always represents control from one party's perspective, choreography tracks message sequences among multiple parties from a global point of view (Peltz, 2003).

A choreography can be specified graphically in modeling notations such as UML (OMG, 2007) and BPMN (Object Management Group, 2009a). A well known example is the RosettaNet Partner Interface Processes (PIPs) (RosettaNet Program Office, 2008) which are modeled in UML. The ebXML BPSS is an XML-based language but it has been described using UML Activity Diagram concepts such as start state, completion state, activities, synchronizations, transitions between activities, and guards on the transitions (OASIS, 2006).

The Web Services Choreography Description Language (WS-CDL) (Kavantzas et al., 2005) is the latest effort spearheaded by the W3C to provide a XML-based choreography specification language specifically for web services enabled collaborative processes. It, too, can be described in UML Activity Diagram (Barros, Dumas, & Oaks, 2006) and other modeling notations such as BPMN (Object Management Group, 2009a), although the WS-CDL standard does not specify nor rely on any such notations.

In practice, business analysts are not expected to express business protocols directly in WS-CDL; instead, they use visual notations such as UML and graphical user interface (GUI) tools for modeling these protocols. However, the use of visual notations in modeling web services choreography has so far been done in an ad hoc fashion as seen in the literature (e.g. see (Barros et al., 2006)); there has been no systematic research on how we can exploit existing standard visual modeling notations for web services choreography that takes into account fully the semantics of WS-CDL. The aim of our research is to establish such a visual approach to modeling web services choreography. In this paper, we present some initial results of our research.

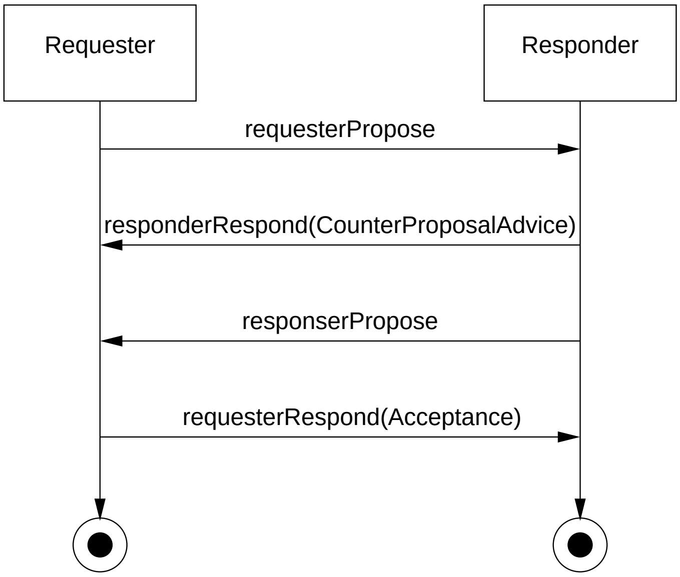

Figure 1: A scenario of the contract negotiation process

This paper surveys and compares a number of commonly used visual modeling notations for modeling web services choreography. They include the UML Sequence Diagram (SD) and Activity Diagram (AD), statecharts (Harel, 1987) and BPMN. A choreography example based on a contract negotiation protocol is used for illustrating the similarities and differences in these notations with regard to choreography modeling.

The next section describes a WS-CDL example based on a contract negotiation protocol adapted from (Rebstock, Thun, & Tafreschi, 2003). Section 3 presents and compares the visual modeling of the protocol in four different notations. Section 4 concludes the paper and indicates some further work.

2 A CONTRACT NEGOTIATION PROTOCOL IN WS-CDL

To illustrate the use of WS-CDL in specifying e-negotiation protocols, the contract negotiation protocol previously discussed in (Rebstock et al., 2003) is adapted for use in here as the main example. The protocol involves two participants, namely Requester and Responder, in an indefinite cycle of sending contract proposals and responses to each other in an alternate fashion. The process begins with the Requester sending an initial proposal to the Responder and ends as soon as either participant issues an acceptance or rejection reply. Figure 1 shows a scenario of the process.

A WS-CDL specification is used for specifying the pattern of web service invocations among the participants in a collaborative process. There is no restriction on the number of participants; a participant that provides services to other participants may at the same time invoke services of the others. Figure 2 shows the structure of a WS-CDL specification. The pattern of web service invocations is specified in one or more choreographies (in the Choreography-Notation section) in terms of *activities*. There are three types of activities, namely *control-flow activities*, *workunit activities,* and *basic activities*. The control-flow and workunit activities are structuring constructs that include sequence, parallel, and choice which correspond the usual programming constructs; the workunit construct renders an activity, or the repetition of it, conditional upon an Boolean expression. There are four kinds of basic activities:

```
<package
        name=...
        author=...
        version=...
        targetNamespace=...
        xmlns="http://www.w3.org/?
        2004/04/ws-chor/cdl">
        importDefinitions*
        informationType*
        token*
        tokenLocator*
        roleType*
        relationshipType*
        participantType*
        channelType*
        Choreography-Notation*
</package>
```

Figure 2: Structure of a WS-CDL specification

- An interaction activity refers to an exchange of information between participants. An interaction may involve either a request, a response, or a request-response pair of information exchange.
- A perform activity “calls” a separately defined choreography.
- An assign activity assigns the value of one variable to another variable within a participant.
- A noaction activity does nothing.

The initial interaction activity involves the Requester’s sending a proposal document to the Responder and then the latter returning a response to the former. Here is the corresponding segment in the WS-CDL specification:

```
 <interaction name="InitialProposal"
         channelVariable="proposalChannel"
        operation="receiveNP">
<participate
        relationshipType="RequesterResponder"
        fromRoleTypeRef="Requester"
        toRoleTypeRef="Responder"/>
<exchange name="requesterRequest"
         informationType="negotiableProposalType"
         action="request">
        <send variable="np"/>
        <receive variable="np"/>
 </exchange>
<exchange name="responderRespond"
        informationType="npResponseType"
        action="request">
        <send variable="response"/>
        <receive variable="response"/>
</exchange>
<timeout time-to-complete="...">
</interaction>
```

Every interaction is defined with a name (line 1) and a channel (line 2) through which information is exchanged. The definition of a channel includes, among other things, operations upon the exchanged information and a particular one is designated for the interaction (line 3). The participants of the interaction are identified by their roles (lines 6 & 7). Two roles may relate with each other in more than one way and a particular way of relationship has to be identified for this interaction (line 5). The information exchanged in an interaction has its type specified (line 9) and also the direction of the information flow (line 10). Variables used by the roles in sending and receiving information are also specified (lines 11 & 12). Note that a timeout condition is placed on the completion of the initial interaction as shown in line 20—if the interaction cannot be completed within a certain time period, an exception handler (see below) will take over the control.

After the initial interaction activity, subsequent interactions is conditional upon the response from the Responder. This is defined in a *workunit* as follows:

```
<workunit name="RequesterProposalWU"
        guard="cdl:getVariable('response')='CounterProposalAdvice'">
        <interaction name="RequesterProposal" ...>
        ...
        </interaction>
 </workunit>
```

Note that the guard condition refers to the variable response. The definition of RequesterProposal is just similar to the InitialProposal interaction above.

The overall choreography involves the indefinite alternation of the REQUESTER PROPOSAL and RESPONDER PROPOSAL interactions, which is outlined in Figure 3. Note that when a timeout does occur, the exception block defined with the current choreography (lines 49-54) is invoked.

Note that repetition of the Alternation workunit is governed by the repeat condition (lines 22-23). The definition of the ResponderProposal interaction is also similar to InitialProposal and is omitted here. The definition of the variable response can also be found in the above choreography definition (lines 6-8).

```
<choreography name="ContractNegotiation"
              root="true>
<relationship type="RequesterResponder" />
<relationship type="ResponderRequester" />
<variableDefinitions>
<variable name="response"
informationType="responseType"
roleTypes="Requester Responder" />
</VariableDefinitions>
<sequence>
<interaction name="InitialProposal"
channelVariable="proposalChannel"
operation="receiveNP">
<participate
relationshipType="RequesterResponder"
fromRoleTypeRef="Requester"
      toRoleTypeRef="Responder"/>
<exchange name="requesterRequest"
informationType="proposalType"
  action="request">
<send variable="np"/>
<receive variable="np"/>
</exchange>
<exchange name="responderRespond"
informationType="responseType"
action="respond">
<send variable="response"/>
<receive variable="response"/>
</exchange>
<timeout time-to-complete="...">
</interaction>
<workunit name="Alternation"
repeat="cdl:getVariable('response')
='NPCounterProposalAdvice'">
<sequence>
<interaction name="ResponderProposal" ... >
...
</interaction>
<workunit name="RequesterProposalWU"
guard="cdl:getVariable('response')
='NPCounterProposalAdvice'">
<interaction name="RequesterProposal" ...>
...
</interaction>
</workunit>
</sequence>
</workunit>
</sequence>
<exceptionBlock name="handleTimeoutException>
<workunit name="handleTimeout"
guard=
    "cdl:hasExceptionOccurred('TimeoutException',..)">
...
</exceptionBlock>
</choreography>
```

Figure 3: A WS-CDL choreography for the contract negotiation process

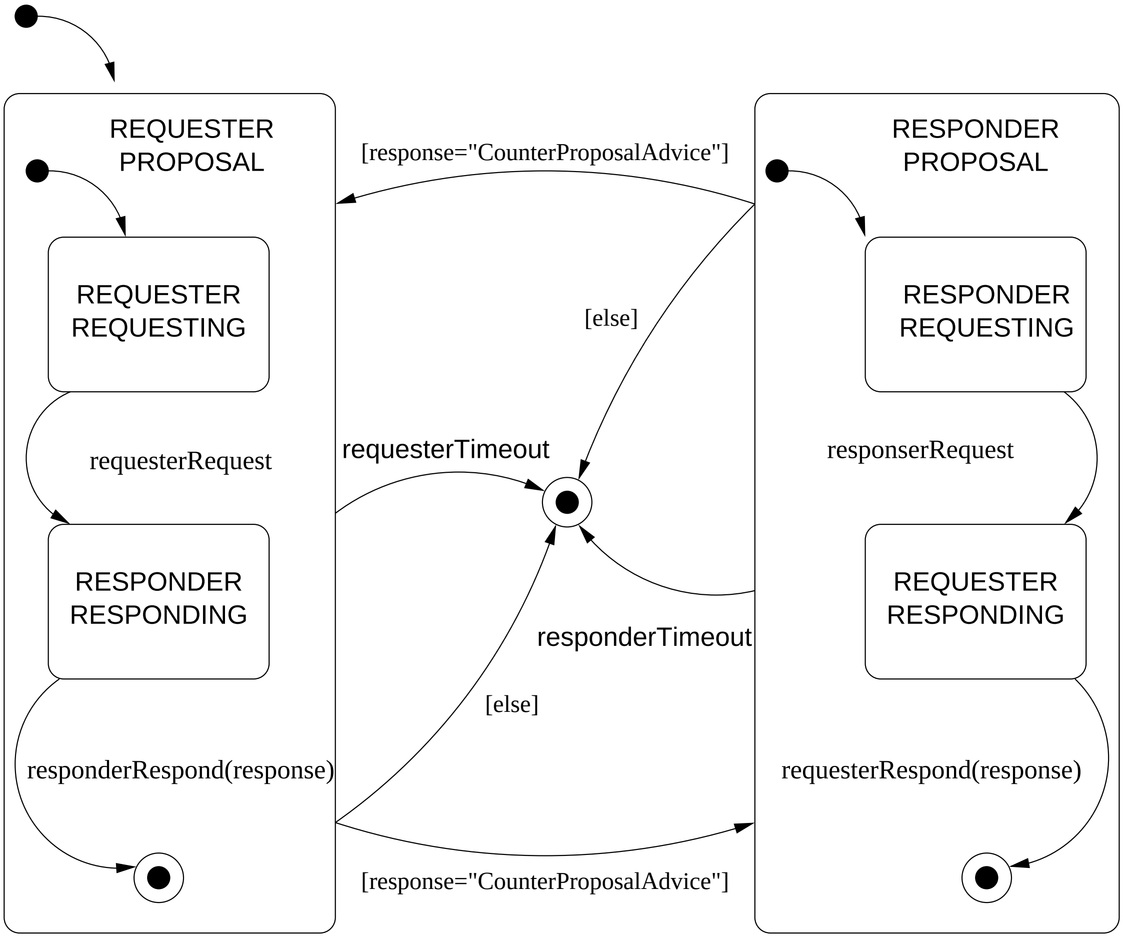

Figure 4: Statechart diagram of a contract negotiation protocol

3 VISUAL MODELING APPROACHES COMPARED

Figure 4,5 & 6 show the modeling of the contract negotiation protocol in statecharts, UML Activity Diagram and BPMN, respectively. Table 1 compares the three different approaches to modeling the protocol in relation to the semantics of WS-CDL. Note that we have also added a fourth approach based on UML Sequence Diagram (as the one seen in Figure 1) in our comparison.

The concept of a role is explicit in a choreography in WS-CDL and it allows the role(s) of every participant involved in collaborative process to be clearly defined. Relationships between two roles are also explicitly defined in WS-CDL. Among the four visual modeling approaches, all except statecharts support the explicit modeling of roles while none of them models relationships explicitly. Even though relationships can be deduced from the presence of interactions between roles, explicit declaration of relationships in WS-CDL can provide a form of static checking for the interactions.

Interactions are expressed as a form of message passing in UML SD and BPMN whereas they correspond to events and activities, respectively, in statecharts and UML AD. Note that UML AD adopts the view that a “request-reply” interaction takes place as an activity on the receiver side as advocated in (Barros et al., 2006). An activity can also be annotated with the types of messages exchanged as shown in Figure 5. On the other hand, the use of events in statecharts for modeling interactions does away with the direction of message flow and simply registers the occurrences of request and reply events without referring to any roles explicitly.

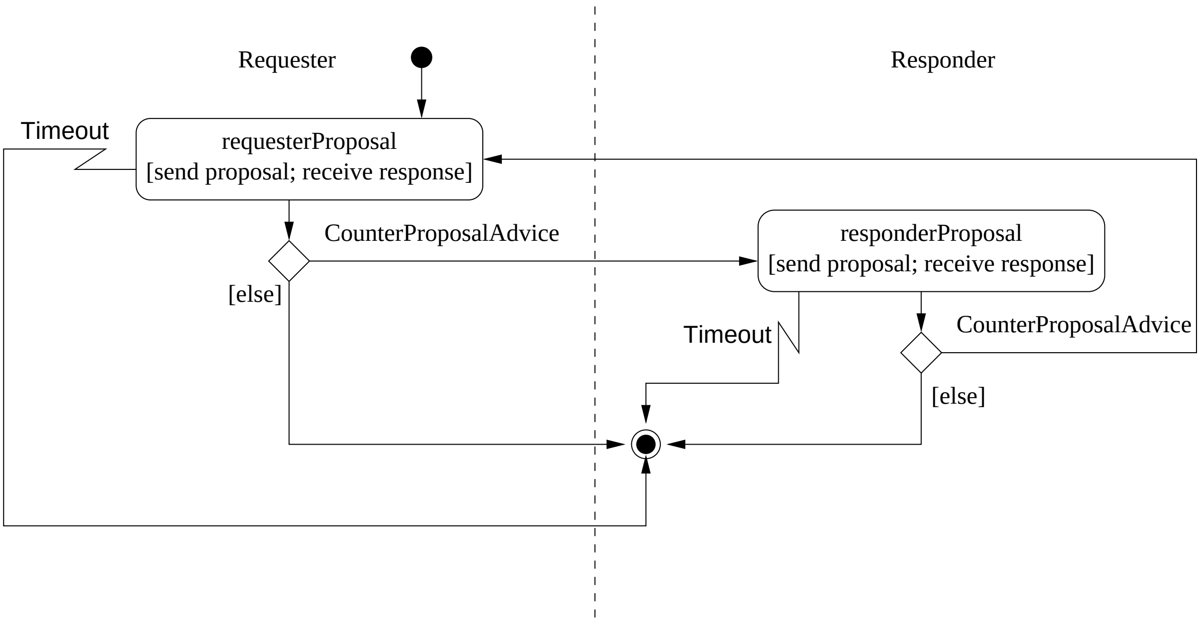

Figure 5: UML activity diagram of a contract negotiation protocol

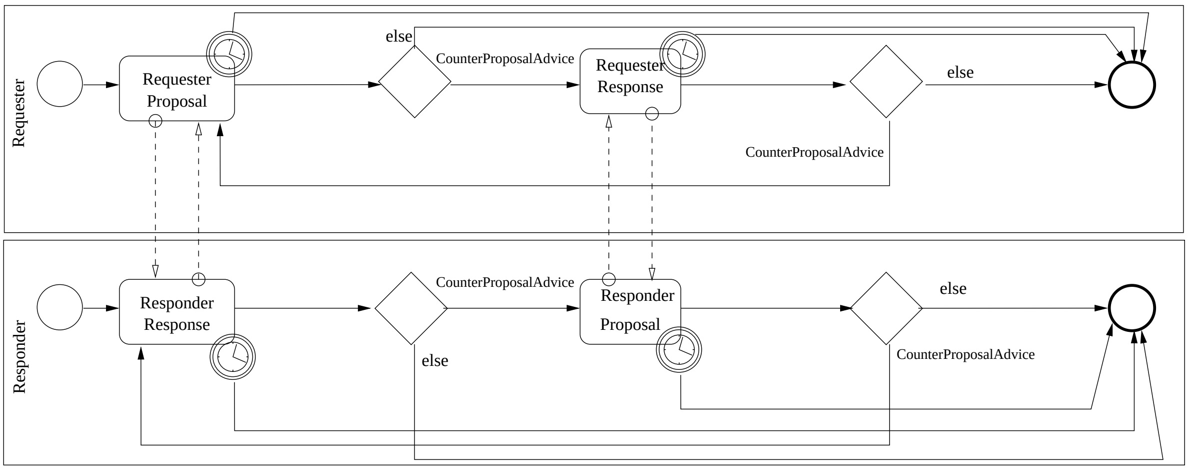

Figure 6: BPMN diagram of a contract negotiation protocol

Flow control is also handled differently among the modeling approaches. UML SD, AD and BPMN use flowchart-like graphs for sequential and parallel flow control. In statecharts, flow control is effected by conventional states and transitions and enhanced by hierarchical (composite) states, implicit transitions (for exception handling) and concurrent regions (for parallel flow). With exit actions, we can even model choreography finalization in state-charts whereas there is no explicit way of doing that in the other modeling approaches.

Finally, there is no provision for the use of channel passing in any of the modeling approaches considered here; in fact, the concept of a channel is not explicit in these approaches.

Table 1: Comparison of visual modeling approaches

	Sequence Diagram	Statecharts	Activity Diagram	BPMN
Explicit roles	Yes	No	Yes (swimlanes)	Yes
Explicit relationships	No	No	No	No
Form of interaction	Message passing	Event	Activity	Message passing
Sequential flow	Mixed	States and transitions	Flowchart-like	Flowchart-like
Parallel flow	Yes (fork and join)	Yes (regions)	Yes (fork and join)	Yes (fork and join)
Exception handling	No	Explicit or implicit (triggerless) transitions	Yes	Yes
Finalization	No	Yes (exit actions)	No	No
Channel passing	No	No	No	No

4 DISCUSSION AND RELATED WORK

As the choreography approach gains wider adoption, visual modeling languages are also evolving to encompass the approach more explicitly. For instance, the beta version of BPMN 2.0 (Object Management Group, 2009b) formally incorporates the choreography view into business process modeling.

Apart from choreography, visual modeling notations are also applicable to the modeling of web service orchestrations. This would allow the use of a single modeling language for both the orchestration and choreography of web services and thereby facilitating development tasks such as model transformation and conformance checking. In fact, both UML activity diagram and BPMN are commonly used for modeling web service orchestrations (see e.g. (Barros et al., 2006; White, 2005; Ouyang, Dumas, Aalst, Hofstede, & Mendling, 2009)).

The methodology of the comparison presented in this paper is an ad hoc one which is based on author's experience on the visual modeling languages. A more systematic comparison would require certain theoretical underpinnings such as the Bunge-Wand-Weber (BWW) representation model (Wand & Weber, 1990; Wand, Monarchi, Parsons, & Woo, 1995; Webber & Wand, 1995), as demonstrated in a comparison of 12 process modeling techniques (Recker, Rosemann, Indulska, & Green, 2009).

5 CONCLUSION AND FURTHER WORK

We have presented an *ad hoc* comparison of four different approaches to modeling web services choreography based on a contract negotiation protocol example. These modeling approaches are meant to capture visually the semantics of WS-CDL and the results of our comparison highlight their differences. None of these approaches support the semantics of WS-CDL fully; they also vary in the style of expression as well as the extent of supporting individual features of WS-CDL. This suggests that the different approaches can actually complement each other in providing business analysts with a multi-faceted approach to modeling web services choreography.

Further work is needed for more comprehensive comparisons of the different approaches in view of our initial findings. Proper theoretical underpinnings are also needed; so are more elaborate case studies that would help reveal correspondences and mismatches between the visual notations and WS-CDL.

Acknowledgement

The author would like to thank the reviewers for their helpful comments and suggestions.

References

Albrecht, C. C., Dean, D. L., & Hansen, J. V. (2005). Marketplace and technology standards for B2B e-commerce: progress, challenges, and the state of the art. *Information & Management*, *42*(6), 865–875.

Barros, A., Dumas, M., & Oaks, P. (2006). Standards for Web Services Choreography and Orchestration: Status and Perspectives. In *BPM 2005 International Workshops, BPI, BPD, ENEI, BPRM, WSCOBPM, BPS, Nancy, France, September 5, 2005. Revised Selected Papers* (Vol. 3812, pp. 61–74). Springer.

Chen, M., Zhang, D., & Zhou, L. (2007). Empowering collaborative commerce with Web services enabled business process management systems. *Decision Support Systems*, *43*(2), 530–546.

Harel, D. (1987, June). Statecharts: A visual formalism for complex systems. *Science of Computer Programming*, *8*(3), 231–274.

Kavantzas, N., et al. (2005). *Web Services Choreography Description Language Version 1.0.* (http://www.w3.org/TR/ws-cdl-10/)

Kim, J. B., & Segev, A. (2005). A Web Services-enabled marketplace architecture for negotiation process management. *Decision Support Systems*, *40*, 71–87.

Lim, B., & Wen, H. J. (2003, Spring). Web services: An analysis of the technology, its benefits, and implementation difficulties. *Information Systems Management*, 49–57.

OASIS. (2006, December). *EbXML Business Process Specification Schema 4 Technical Specification V2.0.4.* Available from http://docs.oasis-open.org/ebxml-bp/2.0.4/OS/spec/ebxmlbp-v2.0.4-Spec-os-en.pdf. ([Accessed on 10 February 2010])

Object Management Group. (2009a, February). *Business Process Modeling Notation (BPMN) Version 1.2.* Available from http://www.omg.org/docs/formal/09-01-03.pdf ([Accessed on 10 February 2010])

Object Management Group. (2009b, August). *Business Process Modeling Notation (BPMN) Version 2 (Beta).* Available from http://www.omg.org/spec/BPMN/2.0 ([Accessed on 9 April 2009])

OMG. (2007, November). OMG Unified Modeling Language (OMG UML), Superstructure, V2.1.2. Available from http://www.omg.org/spec/UML/2.1.2.Infrastructure/PDF (Accessed on 3 February 2010))

Ouyang, C., Dumas, M., Aalst, W. M. P. V. D., Hofstede, A. H. M. T., & Mendling, J. (2009). From business process models to process-oriented software systems. *ACM Trans. Softw. Eng. Methodol.*, *19*(1), 1–37.

Peltz, C. (2003, October). Web services orchestration and choreography. *Computer*, 46-52.

Rebstock, M., Thun, P., & Tafreschi, O. A. (2003). Supporting Interactive Multi-Attribute Electronic Negotiations with ebXML. *Group Decision and Negotiation*, *12*, 269-286.
Recker, J., Rosemann, M., Indulska, M., & Green, P. (2009). Business Process Modeling-A Comparative Analysis. *Journal of the Association for Information Systems*, *10*(4), 333–363.
RosettaNet Program Office. (2008, April). *RosettaNet Overview: Clusters, Segments, and PIPs Version 02.04.00.* Available from http://www.rosettanet.org/cms/export/sites/ default/RosettaNet/Downloads/RStandards/ ClustersSegmentsPIPsOverview 23April2008.pdf ([Accessed on 3 February 2010])
Wand, Y., Monarchi, D., Parsons, J., & Woo, C. (1995). Theoretical foundations for conceptual modelling in information systems development. *Decision Support Systems*, *15*(4), 285–304.
Wand, Y., & Weber, R. (1990). An Ontological Model of an Information System. *IEEE Transactions on Software Engineering*, *16*(11), 1292.
Webber, R., & Wand, Y. (1995). On the deep structure of information systems. *Information Systems Journal*, *July*, 203–23.
White, S. A. (2005). Using BPMN to model a BPEL process. *BPTrends*, *6*, 1–18.
Zhao, J. L., & Cheng, H. K. (2005). Web services and process management: A union of convenience or a new area of research? *Decision Support Systems*, *40*, 1–8.