UML Modelling of Automated Business Processes with a Mapping to BPEL4WS

Tracy Gardner

IBM UK Laboratories, Hursley Park, Winchester, SO21 2JN, UK
tgardner@uk.ibm.com

Abstract. The Business Process Execution Language for Web Services (BPEL4WS) provides an XML notation and semantics for specifying business process behaviour based on Web Services. A BPEL4WS process is defined in terms of its interactions with partners. A partner may provide services to the process, require services from the process, or participate in a two-way interaction with the process.
The Unified Modeling Language™ (UML™) is a language, with a visual notation, for modeling software systems. The UML is an OMG™ standard and is widely supported by tools. UML can be customized for use in a particular modeling context through a 'UML profile'. We describe a UML Profile for Automated Business Processes which allows BPEL4WS processes to be modeled using an existing UML tool. We also describe a mapping to BPEL4WS which can be automated to generate web services artifacts (BPEL, WSDL, XSD) from a UML model meeting the profile.

1 Introduction

As service-oriented technology gains in popularity, it will be increasingly necessary to be able to design large-scale solutions that incorporate web services. The Unified Modeling Langauge™ (UML™) is widely used in the development of object-oriented software and has also been used, with customizations, for component-based software, business process modelling and systems design. UML provides a visual modeling notation which is valuable for solution design and comprehension. UML can be customized to support the modelling of systems that will be completely or partially deployed to a web services infrastructure. This enables the considerable body of UML experience to be applied to the maturing web services technologies. This paper introduces a UML profile (a customization of UML) which supports modelling with a set of semantic constructs that correspond to those in the Business Process Execution Language for Web Services[1] (BPEL4WS)[1].

Using UML primarily as a documentation tool has a real but limited benefit, and it is recognized that UML models developed for this purpose may not be maintained when a project is under severe time pressure. The value of UML-modelling of systems has the potential to increase significantly through the emergence of initiatives such as model-driven development and architected RAD [3]

[1] The current version of the profile is based on BPEL4WS version 1.0.

which enable executable systems to be generated automatically from detailed models. This approach is employed here to provide a mapping from models conforming to the UML profile for automated business processes to executable BPEL processes.

2 The UML Profile for Automated Business Processes

This section introduces a subset of the UML profile through an example that defines a simple purchase order process. A complete specification of the profile can be found in [2]. The example used here is taken from the BPEL 1.0 specification:

"On receiving the purchase order from a customer, the process initiates three tasks in parallel: calculating the final price for the order, selecting a shipper, and scheduling the production and shipment for the order. While some of the processing can proceed in parallel, there are control and data dependencies between the three tasks. In particular, the shipping price is required to finalize the price calculation, and the shipping date is required for the complete fulfillment schedule. When the three tasks are completed, invoice processing can proceed and the invoice is sent to the customer."

BPEL processes are stateful and have instances so in BPEL this scenario is implemented as a PurchaseOrder process which would have an instance for each actual purchase order being processed. Each instance has its own state which is captured in BPEL variables. In the UML profile, a process is represented as a class with the stereotype <<Process>>. The attributes of the class correspond to the state of the process (its containers in BPEL4WS 1.0 terminology). The UML class representing the purchase order process is shown in Figure 1.

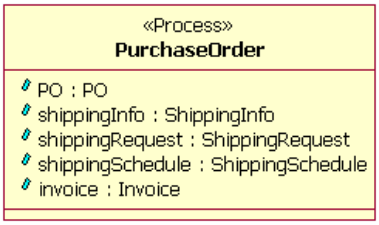

Fig. 1. A UML class used to model a BPEL process

The behaviour of the class is described using an activity graph. The activity graph for the purchase order process is shown in Figure 2. The partners with which the process communicates are represented by the UML partitions (also known as swimlanes): customer, invoiceProvider, shippingProvider and schedulingProvider. Activities that involve a message send or receive operation to a partner appear in the corresponding partition. The arrows indicate the order in which the process performs the activities.

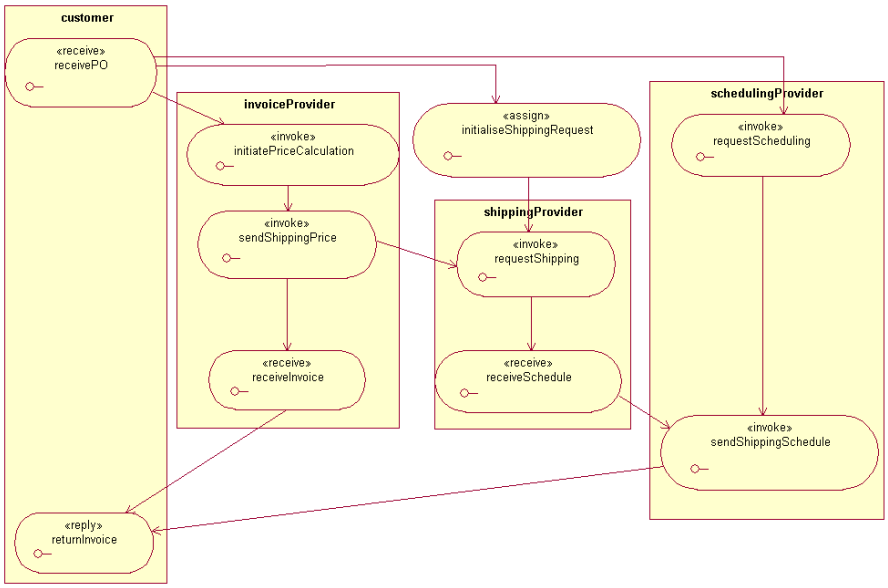

Fig. 2. Activity graph for the purchase order process with detail elided

The purchase order process begins by receiving a purchase order request from a customer. The initiatePriceCalculation, initialiseShippingRequest and requestProductionScheduling activities begin executing, triggering further activities as they complete. The arrows on the graph indicate control links, an activity starts when all of its preceding activities have completed. Note that the requestShipping activity requires that both the initialiseShippingRequest and sendShippingPrice activities have taken place before it begins. The returnInvoice activity returns a response back to the customer. Each activity has a descriptive name and an entry action detailing the work performed by the activity. Note that in Figure 2, the detail of the actions is hidden on the diagram due to space constraints. For a full explanation of the detailed expression of actions please refer to [2].

3 Mapping to BPEL4WS

The UML profile for automated business processes is sufficiently expressive that complete executable BPEL4WS artifacts can be generated from UML models. Table 1 shows an overview of the mapping from the profile to BPEL4WS (version 1.0) covering the subset of the profile introduced in this paper.

A cutdown version of the BPEL document that would be generated from the purchase order example in this paper is shown in Figure 3 (much of the detail is omitted here due to space constraints).

Table 1. UML to BPEL4WS mapping overview.

<<Process>> class	BPEL process definition
Activity graph on a <<process>> class	BPEL activity hierarchy
<<process>> class attributes	BPEL containers
Hierarchical structure and control flow	BPEL sequence and flow activities
<<receive>>, <<reply>>, <<invoke>> activities	BPEL receive, reply, invoke activities

```
<process name="purchaseOrderProcess" ...>
   <containers>
      <container name="PO" messageType="lns:POMessage"/>
      <container name="Invoice" messageType="lns:InvMessage"/>
      ...
   </containers>
   ...
   <sequence>
      <receive partner="customer"
               portType="lns:purchaseOrderPT"
               operation="sendPurchaseOrder"
               container="PO">
      </receive>
      ...
      <reply partner="customer" portType="lns:purchasePT"
             operation="sendPurchaseOrder"
             container="Invoice"/>
   </sequence>
</process>
```

Fig. 3. BPEL extract corresponding to the purchase order process.

4 Proof of Concept Demonstrator

A technology demonstrator supporting an end-to-end scenario from a UML tool (such as Rational™ XDE™) through to a BPEL4WS runtime (BPWS4J) is available from IBM™ alphaWorks™ as part of the Emerging Technologies Toolkit [4]. The mapping implementation is built using the Eclipse Modeling Framework (EMF) and takes the industry standard file format for exchange of UML models (XMI) as input. BPEL4WS artifacts along with the required WSDL and XSD artifacts are generated.

5 Conclusion

This paper has introduced a UML profile for automated business processes with a mapping to BPEL4WS. This approach enables service-oriented BPEL4WS components to be incorporated into an overall system design utilizing existing soft-

ware engineering practices. Additionally, the mapping from UML to BPEL4WS permits a model-driven development approach in which BPEL4WS executable processes can be automatically generated from UML models. A proof of concept demonstrator for the mapping is available. Future work includes the implementation of a reverse mapping to support the import of existing BPEL4WS artifacts and the synchronization of UML models and BPEL4WS artifacts with changes in either being reflected in the other. The profile and mapping currently support the 1.0 version of the BPEL4WS specification, support for BPEL4WS 1.1 is planned.

Acknowlegements

Thanks to Gary Flood and Keith Mantell for comments on this paper.

References

1. BEA Systems, IBM, Microsoft: Business Process Execution Language for Web Services, Version 1.0. IBM developerWorks (2002). Available from http://www-106.ibm.com/developerworks/webservices/library/ws-bpel1/
2. Gardner, T et al.: Draft UML 1.4 Profile for Automated Business Processes with a mapping to the BPEL 1.0. IBM alphaWorks (2003). Available from http://dwdemos.alphaworks.ibm.com/wstk/common/wstkdoc/services/demos/uml2bpel/README.htm.
3. Selic, B: The Pragmatics of Model-Driven Development. IEEE Software special issue on Model-Driven Architecture (2003). (To be published.)
4. Emerging Technologies Toolkit. IBM alphaWorks (2003). Available from http://www.alphaworks.ibm.com/tech/ettk/