Using UML to Design Distributed Collaborative Workflows: from UML to XPDL

Ping Jiang, Quentin Mair, Julian Newman
Department of Computing, Glasgow Caledonian University, Glasgow G4 0BA, UK
{p.jiang, q.mair, j.newman}@gcal.ac.uk

Abstract

Business process modelling and workflow process execution are often conducted in diverse environments and described using diverse process definition languages. Such systems often underpin distributed collaboration systems, but there is a current need to allow developers to use existing and familiar design methodologies and tools to design these systems. This paper presents the business model architecture used in the DIECoM[1] project and examines the problem of how to transfer multiple views on a business process model in UML to a computer view for workflow execution. The roles and relationships of various views described by Use Case Diagrams, Sequence Diagrams, Statechart Diagrams and Activity Diagrams are clarified and the missing information is supplied to facilitate the design of a uniform executable workflow model. As a result, the process models defined in the proposed way are consistent with XPDL semantics and can be readily translated to an XPDL file with the aid of an XSLT processor.

1. Introduction

One of the basic objectives of the DIECoM project is to develop a generic business model to support collaborative Product Configuration Management (PCM) across heterogeneous tools and virtual organizations. The generic business model can be thought of as a set of views into the PCM system being developed from different perspectives. Unified Modeling Language (UML) has been chosen to support business process modelling in the DIECoM project – it is familiar to most professional software designers. After the generic business model is defined in UML, significant issues arise about how the defined business process can be transferred to the various available commercial workflow management systems for DIECoM implementation. Usually, commercial workflow applications can support distributed integration via a modelling standard. Unfortunately, several parallel standards exist today, and these are funded by different consortia or standards bodies for different application fields:

- NIST Process Specification Language (PSL)[1]
- DARPA Agent Markup Language for the semantic web (DAML-S)[2]
- OMG UML Extensions for Workflow Process Definition (RFP)[3]
- WfMC Workflow Process Definition (XPDL) for distribution of workflow[4]
- BPMI Business Process Modeling Language (BPML) for web services[5]
- IBM/Microsoft Business Process Execution Language (BPEL) for web services[6]

Although all of these claim that they are *the* standard for process modelling, no single standard has been able to convince everyone to adopt it. As a result, process model interchange and reuse in distributed systems has not been very successful. Currently, these process modelling standards adopt different approaches. Even the adoption of standard XML syntax does not standardize process semantics definitions; different approaches such as block-based or graph-based[7] representations yield different XML-based standards. A properly designed XSL (eXtensible Stylesheet Language) stylesheet for transforming an XML process model from one standard to another can be a feasible way to achieve the heterogeneous system integration. This paper reports a case study of transforming a process model in UML to XPDL using an XSL stylesheet.

In the DIECoM project, Rational Rose is selected as a UML tool for process modelling. The model should then be transferred to DS/IBM ENOVIA LCA for enactment purposes. Both Rational Rose and ENOVIA have considered how to integrate with other tools. Rational Rose can export its models to XMI format with the help of a freely available XMI addin. ENOVIA, for its

[1] DIECoM (Distributed Integrated Environment for Configuration Management) is an IST project funded under the EC's Framework V programme. DIECoM runs from September 2001 to August 2003. The project members are:- Alenia, Dassault Systemes, EADS-CCR, IBM, Glasgow Caledonian University, Renault and SIA. See **http://www.diecom.org**.

Proceedings of the Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE'03)
1080-1383/03 $17.00 © 2003 IEEE

workflow enactment, can accept business process templates from any other WfMC compliant workflow application using XPDL templates. Although the UML Activity Diagram has been proposed to describe business processes, its semantics are left largely unspecified for business process modelling[8]. Therefore, simply defining a process by using Activity Diagrams in UML cannot include all the precise information required by XPDL for workflow execution. Paper [9] extended UML Activity Diagrams for workflow modelling. It associated the Activity Diagram elements with the XPDL C-Wf classes using stereotypes and defining some new properties to provide information for workflow definition. Paper [10] pointed out that no single type of UML diagram captures all of the information needed to describe a process.

This paper presents an XSLT based model transformation from business process models in UML to workflow models in XPDL. Firstly, the organizational architecture of business process model in the DIECoM project is introduced. The multiple views of the business model are then compared with the workflow meta-model defined in XPDL. An XPDL compliant business process model composed of UML diagrams is proposed to aid the model translation. The detailed entities mapping is defined and the lack of details in UML for workflow definition is supplied by using stereotypes, events, etc. The resulting business process model saved using Rational Rose into XMI format can then be transferred to XPDL for workflow execution by means of stylesheet description.

2. UML generic business model

A business process is a logical structure of people, technology, and practices that are organized into work activities designed to transform information, materials, and energy into specified end result(s). Therefore, a business process model is an abstraction of a real or conceptual complex system. The model should be designed to display significant features and characteristics of a PCM system, which the DIECoM project wishes to study, predict, modify, and control, i.e. an integration of Product Data Management (PDM), Software Configuration Management (SCM), and Electrical Data Management (EDM).

The DIECoM business model is organized in a 3-level hierarchical structure, i.e. organization, coordination and execution. An overall business model structure can be described by Figure 1. On the organizational level, UML Package diagrams are used to model organization mode and domains. At this level the model declares what organization units and groups are involved in a virtual enterprise for their common objectives and individual

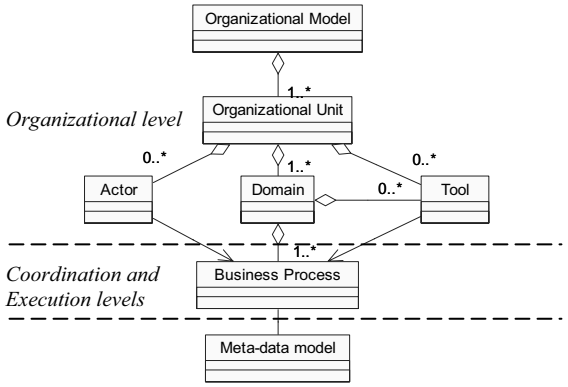


Figure 1. DIECoM business model

responsibility. As the mission of the DIECoM project is to develop an integration environment for cross-organizational product management, the configuration management domains include PCM, SCM, EDM and PDM, which basically defines a set of business objectives and constraints. The involved actors and tools are also described by a Main Actors Use Case Diagram and a Tool Hierarchy Class Diagram. A business process which interacts with the DIECoM Meta-data model to accomplish part or whole domain objectives under given constraints can then be defined. The next two levels, coordination and execution, refine the business process and attempt to turn general business requirements into an executable computer specification. Business processes are represented by use cases and use case instances on the coordination level and the execution level, respectively. The use cases at the coordination level are definitions of business processes in terms of goals, responsibilities, preconditions and postconditions, which are modelled by

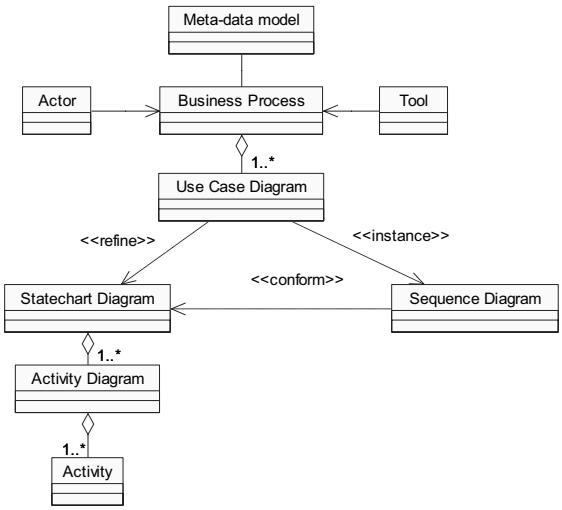


Figure 2. DIECoM coordination and execution levels definition for business processes

Use Case Diagrams in UML to represent static relationships between use cases, actors and relevant tools. The use case instances of business processes are concrete sequences of events defining the dynamic behaviour of a business process. This is described by Sequence Diagrams, Statechart Diagrams and Activity Diagrams at the third level, i.e. execution level that is the core for automated business process, i.e. workflow. The 3 different kinds of diagrams at this level are not orthogonal in defining the dynamic performance of the business model. All 3 diagrams are related to each other and aim to describe the business from different viewpoints and in different detail. The Sequence Diagrams describe only typical scenarios and cannot represent all allowable sequences of use case instances. The allowable order of use case instances belonging to a Use Case Diagram can be specified by a Statechart Diagram and further detailed by Activity Diagrams. These are called the life cycle of the use case package[11]. In summary, the DIECoM business process model can be illustrated in Figure 2.

3. Process definition in accordance with XPDL format

Figure 2 clearly illustrates the relationship between different diagrams for representing business processes. In the model, the Statechart Diagram and Activity Diagram define the allowable order of use case instances belonging to a specific use case package. They define a precise representation of the use case package behaviour (the lifecycle of the use case package). In contrary, the Sequence Diagram describes only typical scenarios. It should conform to the generic workflow definition represented by the Statechart and Activity Diagrams. In other words, the designed Statechart and Activity Diagrams should be designed to cover all possible scenarios represented by the Sequence Diagrams. The reason for defining specific sequences in a business model is that a generic and overall definition of the business process is usually more difficult than a specific scenario definition. Specific scenarios described by Sequence Diagrams can act as inputs or templates for generic workflow definition.

In the DIECoM project, a Statechart Diagram owned by a Use Case Diagram collects all discrete stages of a workflow lifecycle. The Activity Diagram owned by each state defines the dynamic performance of each state and may drive one state change to another state. Therefore, a Statechart Diagram defines the state space of a use case and Activity Diagrams refine states for execution. Figure 3 shows the relationship between Statechart Diagram and Activity Diagram in the DIECoM scope.

XPDL is a workflow process definition language defined by the Workflow Management Coalition

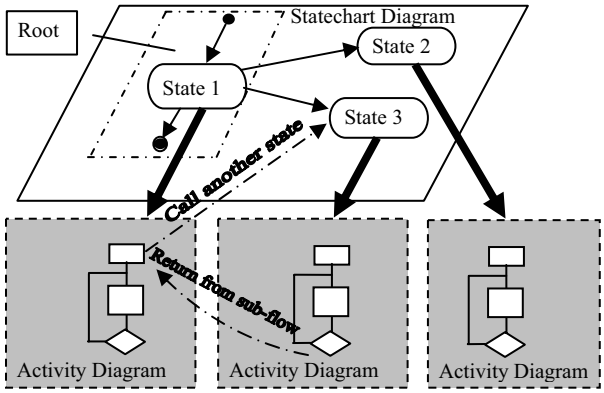


Figure 3. Statechart and Activity Diagram

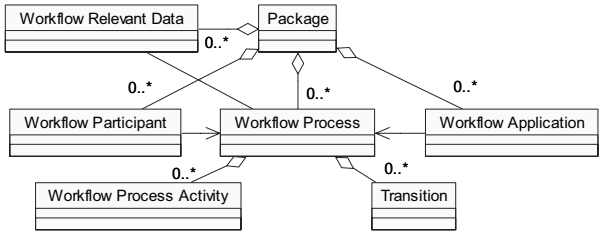


Figure 4. XPDL workflow process model

(WfMC), which is used in ENOVIA for workflow definition exchange. This requires that the DIECoM business model defined in UML can be built based on conformance to the XPDL template and exported to the XPDL format. There are several entities for the definition of workflow process. Within the entities, Workflow Process Activity and Transition Information are internal entities that are used to define the internal relationship of a workflow. Some other entities, Workflow Participant Specification, Workflow Application Declaration, Workflow Relevant Data are in a scope wider than a single process. A number of processes may refer to these common entities. In XPDL, a package is introduced to provide a container to hold a number of common entities: Workflow Participant Specification, Workflow Application Declaration, Workflow Relevant Data. The template of the process definition in XPDL looks like Figure 4. Comparing Figure 4 with Figure 2, we can make the following concept mapping between the DIECoM business process model and XPDL workflow definition:

Table 1. Relationship between DIECoM model and XPDL process model

DIECoM model	XPDL Concept
Use Case Diagram	Package
Statechart Diagram	Workflow processes
Activity Diagram	Workflow process
Instance of DIECoM meta-data model	Workflow relevant data
Instance of DIECoM actor	Workflow participant
Instance of DIECoM tool	Workflow application

In addition to the above concept mapping, XPDL requires the definition of the other workflow related information for the purpose of process automation. This includes Workflow Process Activity, Transition Information, Workflow Participant Specification, Workflow Application Declaration, Workflow Relevant Data, etc., and will be defined in the next section.

4. DIECoM workflow entities specification

After representing a workflow process by composite states and further by Activity Diagrams, the process meta-model needs to be refined with more information. In XPDL, the meta-model requires to define Workflow Process Activity, Transition Information, Workflow Participant Declaration, Workflow Application Declaration and Workflow Relevant Data as shown in Figure 4. Unfortunately, the DIECoM workflow processes definition in UML cannot provide the precise semantic information required by XPDL. Therefore, we need to define stereotypes and assign some attributes with precise meanings so that workflow process model required by XPDL grammar can be equivalently defined by an Activity Diagram in UML without missing details and omitting meaning. The mapping between Activity Diagram elements and XPDL workflow entities can be defined in Table 2

Table 2. Activity Diagram elements mapping to XPDL

Activity Diagram elements in UML	XPDL entities
Activity	Workflow process activity
Transition	Transition information
Swimlane	Workflow participant
Activity with "Tool" stereotype	Workflow application declaration
On Entry "IN" action and On Exit "Out" action	Workflow relevant data

The details of XPDL entities defined in Activity Diagram are proposed below:

4.1. Workflow process activity

A process definition consists of one or more activities.

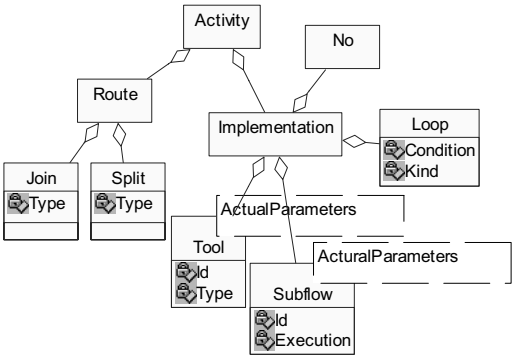


Figure 5. Implementation types of an activity

An activity is a work unit in the process and defines details for workflow execution. Different implementation types of an activity have different attributes as shown in Figure 5.

Therefore, we define the following stereotypes to distinguish activities with different implementation types:

<<Route>>

ENOVIA requires a dummy activity for the implementation of *split* or *join* transitions. The Route activity is a "dummy" activity with attribute "Route" for this purpose. It permits the expression of "cascading" transition conditions. A route activity has neither a performer nor an application and its execution has no effect on workflow relevant data or application data. There are Join Route and Split Route as the routing types. Depending on transitions, the "Transition Restriction" attribute of a route should be assigned with a type of either AND or XOR.

<<No>>

"No" means that it is a manual activity performed by human actors. As a result, the activity attribute "Finish Mode" is set to manual mode that requires explicit user interaction to cause activity finish. As a participant of *no* implementation activity, the performer should be always of the HUMAN type.

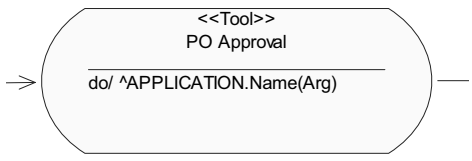
<<Tool>>

An activity with a stereotype of <<Tool>> means that the activity is implemented by one or more application programs. The activity definition in XPDL needs to provide extra information including application name, application type and invocation parameter to the workflow management system as shown in Figure 5. In UML, we define the information using the Action Specification for a given activity as shown in Table 3:

Table 3. Attributes for <<Tool>> activity

XPDL attributes	Mapping attributes in UML
Actual Parameters	(Activity Diagram)->Action-> Do. Send Event. Send arguments
Id	(Activity Diagram)->Action->Do. Send Event. Name
Type (APPLICATION| PROCEDURE)	(Activity Diagram)->Action->Do. Send Event. Send target

An activity example is shown below:



<<Subflow>>

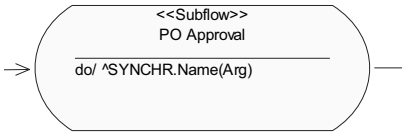
An activity with the <<Subflow>> stereotype is refined as a subflow invocation. The DIECoM composite states in the Statechart Diagram can be <<Subflow>>

activities with their associated Activity Diagram as a body of the subflow. As shown in Figure 5, the subflow associated attributes are Subflow Id, Execution manner and Actual Parameters. We define the following mapping relationship between UML and XPDL using Action Specification in the Activity Diagram.

Table 4. Attributes for <<Subflow>> activity

XPDL attributes	Mapping attributes in UML
Actual Parameters	(Activity Diagram)->Action-> Do. Send Event. Send arguments
Id	(Activity Diagram)->Action->Do. Send Event. Name
Execution (ASYNCHR|SYNCHR)	(Activity Diagram)->Action->Do. Send Event. Send target

An example of the subflow activity is shown below:



<<Loop>>

The activity is refined by a loop body that is defined by the loop body's sub diagram. The loop transition condition is examined in loop activity instead of the transition definition. The condition can be executed in two different ways: "WHILE….DO…" and " REPEAT…UNTIL", which are defined by "loop kind" attribute of a loop activity.

Table 5. Attributes for <<Loop>>

XPDL attributes	Mapping attributes in UML
Loop Body Id	(Activity Diagram)->Action-> On Event. Arguments
Condition	(Activity Diagram)->Action-> On Event. Condition
Kind (WHILE|REPEAT_UNTIL)	(Activity Diagram)->Action->On Event. Event

An example of the Loop activity in UML is shown below:

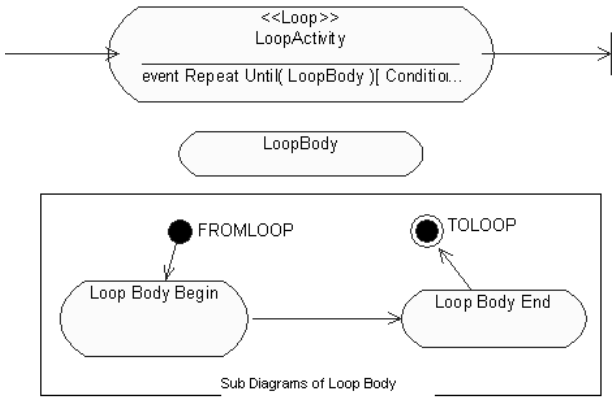


Figure 6. Loop implementation

4.2. Workflow transition

Workflow execution is based on connected activities. The transition information defines the connections and conditions that enable or disable the transition. There are 3 types of transition. A "NOLOOP" type transition represents the regular thread of a workflow and the "FROMLOOP" or "TOLOOP" transitions only connect with a <<Loop>> activity to indicate entry or exit of a loop body as shown in Figure 6. In addition to the transition types, a transition definition should provide information about the connected source and target. It is defined by "From" and "To" attributes of a transition in XPDL. The following table summarizes the defined mapping between UML and XPDL

Table 6. Attributes for workflow transition definition

XPDL attributes	Mapping attributes in UML
Condition	(Activity Diagram)->Transition-> Guard Condition*
From	(Activity Diagram)->Transition-> Source
To	(Activity Diagram)->Transition->Target
Id	Generated by XSLT as (Source-Target)
Loop	Generated by XSLT automatically

*In XPDL, the condition is only available for the <<NOLOOP>> transition. For loop body connections, as indicated in the <<Loop>> activity, the transition condition is controlled by loop activity instead of a <<FROMLOOP>> or <<TOLOOP>> guard condition.

4.3. Workflow participant

Workflow participant defines the actors or applications that can act as performers of various activities. There are 6 types of workflow participants: resource set, resource, organizational unit, role, human and system. Mapping between UML concept and XPDL workflow participant expression can be defined as shown in Table 7 below.

The participant Id should be an instance of any actors or tools defined in the DIECoM Main Actors Use Case Diagram or the DIECoM Tools Class Diagram.

Table 7. Attributes for workflow participant definition

XPDL attributes	Mapping attributes in UML
Id	(Activity Diagram)->Swimlane-> name
Participant Type	(Activity Diagram)->Swimlane->Class (RESOURCE_SET|RESOURCE|ORGANIZATIONAL_UNIT|ROLE | HUMAN |SYSTEM)

Remark: In the XSL stylesheet developed for the DIECoM project, all participants are assigned with "ROLE" type. The advantage is that we can simply declare a performer at build time, then, at runtime, when instantiating the template into an actual process, the

workflow will be allowed to create a context for the process. There are two types of contexts: Private contexts and Public contexts in the ENOVIA LCA.

4.4. Workflow application

Workflow application declaration is a list of applications or tools required and invoked by the workflow process. Only the Id of the application should be defined. In a Workflow Package, we collect all activities with a <<Tool>> implementation type and take the application or procedure name as associated workflow application declarations within the package scope.

4.5. Workflow relevant data

In a process model, two flows exist simultaneously. These are workflow and dataflow. Both flows interact with each other. Workflow may access, reference and manage the data to form dataflow, and, at the same time, dataflow may change the route that a workflow performs. Workflow relevant data defines all data objects, which are required and processed by the workflow process. In order to represent workflow related dataflow clearly, we use On_Entry action and On_Exit action to represent the "IN" and "OUT" data of a workflow, respectively. The data type is constrained to Basic Type and Declared Type. The Basic Type includes string, float, integer, reference and datetime. The declared type can be any defined DIECoM class in meta-data model. Therefore the mapping of UML definitions to XPDL definitions can be described as in Table 8.

Table 8. Attributes for workflow relevant data

XPDL attributes	Mapping attributes in UML
Id	IN data: (Activity Diagram)->Action->On_Entry.Sendevent.name OUT data: (Activity Diagram)->Action->On_Exit.Sendevent.name
DataType	(Activity Diagram)->Action->On_Exit(|Entry).Sendevent.Sendaugment= (DIECoMDATA|STRING|FLOAT|INTEGER|REFERENCE|DATETIME)

Based on the workflow relevant data, the parameters passed along between processes and subflows can be defined in the Formal Parameters declaration. The scope of the formal parameters is for all workflow processes and subflows in the Workflow processes definition. In the DIECoM process model structure defined in Sec.3, any process or subflow was represented by a composite state in the Statechart; therefore the Formal Parameters declaration should be given for all composite states in a Statechart Diagram using the On_Entry and On_Exit event attributes as defined in Table 8.

5. Conclusions

This paper focused on a problem which has arisen from the fact that different XML model formats are accepted by Rational Rose and ENOVIA in DIECoM project, where XMI can be generated by Rational Rose's XMI addin and the WfMC reference model (specifically XPDL) can be accepted by ENOVIA's workflow management tool. In order to integrate UML and WfMC technologies for business processes, the translating method and the mapping relationship were proposed in this paper. Firstly, we established an XPDL compliant process meta model using UML, and then generated the corresponding XMI model for interchange. Consequently, the exported workflow model in XMI format is consistent with XPDL. An XSL stylesheet has been developed to transfer UML process model to XPDL for the ENOVIA LCA workflow environment running. One of the fundamental premises of virtual enterprises is that they must leverage existing methodologies and integrate existing toolsets to have viable uptake. The use of UML is now universal in systems design; we have shown that it is possible to use the industry standard UML tool to design collaborative workflows by extending the semantics of UML. We anticipate that this approach can be extended to generate different representations for different standards e.g. DAML-S.

6. References

[1] PSL, http://ats.nist.gov/psl/.

[2] DAML Services, http://www.daml.org/services/.

[3] UML Extensions for Workflow Process Definition, ftp://ftp.omg.org/pub/docs/bom/00-12-11.pdf.

[4] Workflow process definition interface - XML process definition language - Document Number WFMC-TC-1025, http://www.wfmc.org/standards/docs/TC-1025_10_xpdl_102502.pdf.

[5] Business Process Execution Language for Web Services version 1.1, http://www-106.ibm.com/developerworks/webservices/library/ws-bpel/.

[6] BPML working draft, http://www.bpmi.org/.

[7] R. Shapiro, "A Comparison of XPDL, BPML and BPEL4WS", http://xml.coverpages.org/ni2002-06-26-b.html.

[8] J. Lubell, "XML representation of process descriptions", http://ats.nist.gov/psl/xml/process-descriptions.html.

[9] G.Bruno, M.Torchiano, R.Agarwal, "UML Enterprise Instance Models", Proc. of Int. Conf. on Information Technology (CIT2002), Bhubaneswar, India, 2002.

[10] R. Bastos, D. Dubugras, A. Ruiz, "Extending UML Activity Diagram for Workflow Modeling in Production Systems", 35th Annual Hawaii International Conference on System Sciences (HICSS'02)-Volume 9, 2002.

[11] P. Hruby, "Structuring Specification of Business Systems with UML",OOPSLA'98 Business Object Workshop, 1998.