XPDL 문서 생성을 위한 UML 액티비티 다이어그램의 확장

An Extension of UML Activity Diagram for Generation of XPDL Document

왕보 김재정 유철중 장옥배
Wang Bo[0] Jae-jung Kim Cheol-Jung Yoo Ok-Bae Chang
Software Engineering Lab., Department of Computer Science, Chonbuk National University, Korea
wanglucky[0]@empal.com jajkim@cs.chonbuk.ac.kr {cjyoo, okjang}@moak.chonbuk.ac.kr

Abstract

Currently there are a variety of different tools may be used to analyze, model, describe and document a business process. However, it is difficult to exchange the information of a business process created in different tools because of the distinct information formats used in different tools. The XML Process Definition Language (XPDL) of the Workflow Management Coalition (WfMC) forms a common interchange standard that enables products to continue to support arbitrary internal representations of process definitions with an import/export function to map to/from the standard at the product boundary. Generally a business process model can be represented by the UML activity diagram, but there is a difficult task to directly generate an XPDL document from a business process model represented by the standard activity diagram. In the paper we will propose an approach to generate an XPDL document from a business process model represented by the extended UML activity diagram.

Keyword: XPDL, process definition, UML activity diagram

Introduction

The XPDL specification uses XML as the mechanism for process definition interchange. A Process Definition is defined as: The representation of a business process in a form that supports automated manipulation, such as modeling, or enactment by a workflow management system. The process definition consists of a network of activities and their relationships, criteria to indicate the start and termination of the process, and information about the individual activities, such as participants, associated IT applications and data, etc. [1]. The UML activity diagram also can create business process model in forms of sets of activities and transitions between them [2] [3][4]. So it is possible to map the business process model, which is represented by UML activity diagram, to the process definition organized in XPDL. Hence the paper will propose a method of generating an XPDL document from a business process model represented by an extended activity diagram. For the purpose of this paper, the terms process definition, business process model, and workflow model are all considered to represent the same concept, and therefore, they are used interchangeably.

The paper consists of four sections. The following section discusses issues for related works. The method of a mapping from the business process model represented by extended activity diagram to the XPDL document will be depicted in the third section. Section 4 presents our conclusion and future work.

Issues for related work

Currently a tool for business process modeling has been implemented, which could create extended activity diagram for modeling business process. In order to make the generated business process model available for other modeling tools, there must be a common interchange standard existing for information exchange among the diverse tools. Fortunately XPDL uses XML, which is the current industrial standard of data information organization and exchange, as the mechanism for process definition interchange referred in previous section. Now the key problem is how to generate a XPDL document from a business process model represented by UML activity diagram. The following section will propose a solution.

Mapping from Business Process Model to XPDL Document

Accord to the structure of XPDL document [1], the task of the

mapping actually is to generate the corresponding information in the format of XPDL from each element in source business process model and put the information into the appropriate positions in XPDL document structure. Figure 1 shows a table which shows the associations between the elements of extended activity diagram and entities in XPDL.

Element notation	Description	Corresponding entity in XPDL
	A start state explicitly shows the beginning of a workflow.	Route activity.
	An end state explicitly shows the end of a workflow on an activity diagram.	Route activity.
	An activity with the type of "No Implementation" which is implemented by manual procedures.	A "regular" activity whose implementation type is "No Implementation".
	An activity with the type of "Tool" whose implementation is supported by (one or more) application(s).	A "regular" activity whose implementation type is "Tool"
	An activity with the type of "Subflow" which is Implemented by another process.	A "regular" activity whose implementation type is "Subflow".
	Vertical/Horizontal synchronization define forks and joins representing parallel workflow.	Transition restriction with the attribute of join/spit
	From one state to another state when certain conditions are satisfied	Transition
	A decision represents a specific location on an activity diagram where the workflow may branch based upon guard conditions.	Route activity with transition restriction of spit with XOR type.

Figure 1. The associations between the elements of extended activity diagram and the entities in XPDL

- **The mapping of "start state" and "end state" element**

The "start state" and "end state" element can be mapped to the "Route activity" entity defined in XPDL. The Route Activity is a "dummy" Activity that 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 [1].

Figure 4 shows an example of XPDL representation of "Start state" element. The "Id" attribute is used to identify the activity and the "ExtendedAttributes" attribute are the optional extensions to meet individual implementation needs. The "TransitionRestrictions" attribute will be discussed later.

```
<Activity Id="5">
  <Description>This is a start!</Description>
  <Route/>
  <ExtendedAttributes>
    <ExtendedAttribute Name="Coordinates">
      <xyz:Coordinates xpos="35" ypos="389"/>
    </ExtendedAttribute>
  </ExtendedAttributes>
</Activity>
```

Figure 4. The XPDL representations of "Start state" element

- **The mapping of activity element**

An "Activity" element can be mapped to a "regular" activity entity with one of three different Implementation types, which is defined in XPDL: No Implementation, Tool, and Subflow. "No Implementation" type means that the implementation of this activity is not supported by workflow using automatically invoked applications or procedures. "Tool" type means that the activity is implemented by (one or more) tools. A tool may be an application program. "Subflow" type means that the activity is refined as a subflow. The subflow may be executed synchronously or asynchronously [1]. Figure 5 shows an example of XPDL representation of "Transform Data" activity with "Tool" type.

```
<Activity Id="17" Name="Transform Data">
<Implementation>
  <Tool Id="transformData" Type="APPLICATION">
    <ActualParameters>
      <ActualParameter>orderString</ActualParameter>
      <ActualParameter>orderInfo</ActualParameter>
    </ActualParameters>
  </Tool>
</Implementation>
<TransitionRestrictions>
  <TransitionRestriction>
    <Split Type="XOR">
      <TransitionRefs>
        <TransitionRef Id="40"/>
        <TransitionRef Id="21"/>
      </TransitionRefs>
    </Split>
  </TransitionRestriction>
</TransitionRestrictions>
<ExtendedAttributes>
  <ExtendedAttribute Name="Coordinates">
    <xyz:Coordinates xpos="102" ypos="389"/>
  </ExtendedAttribute>
</ExtendedAttributes>
</Activity>
```

Figure 5 XPDL representation of activity element with "Tool" type.

- ***The mapping of synchronization element***

A synchronization element can be mapped to a transition restriction entity with the attribute of join/spit, which is defined in XPDL. A join corresponding to a join in synchronization describes the semantics of an activity with multiple incoming transitions. A split corresponding to a fork in synchronization describes the semantics where multiple

outgoing transitions for an activity exist. Both join and spit have the types of "AND" and "XOR"[1]. Figure 6 shows an example of XPDL representation of synchronization element.

```
<Activity Id="9">
 <Route/>
 <TransitionRestrictions>
  <TransitionRestriction>
   <Split Type="AND">
    <TransitionRefs>
     <TransitionRef Id="1"/>
     <TransitionRef Id="38"/>
     <TransitionRef Id="2"/>
    </TransitionRefs>
   </Split>
  </TransitionRestriction>
 </TransitionRestrictions>
 <ExtendedAttributes>
  <ExtendedAttribute Name="Coordinates">
   <xyz:Coordinates xpos="572" ypos="389"/>
  </ExtendedAttribute>
 </ExtendedAttributes>
</Activity>
```

(a)

```
<Activity Id="33">
 <Route/>
 <TransitionRestrictions>
  <TransitionRestriction>
   <Join Type="AND"/>
  </TransitionRestriction>
 </TransitionRestrictions>
 <ExtendedAttributes>
  <ExtendedAttribute Name="Coordinates">
   <xyz:Coordinates xpos="725" ypos="391"/>
  </ExtendedAttribute>
 </ExtendedAttributes>
</Activity>
```

(b)

Figure 6 XPDL representation of synchronization element of (a) fork (b) join

- **The mapping of transition element**

A transition element can be mapped to a transition entity defined in XPDL. The transition entities describe possible transitions between activities and the conditions that enable or disable them (the transitions) during workflow execution.[1] Figure 7 shows an example of XPDL representation of transitions element.

```
<Transitions>
   <Transition Id="1" From="9" To="8"/>
   <Transition Id="2" From="9" To="11"/>
   <Transition Id="16" From="11" To="33"/>
   <Transition Id="17" From="8" To="33">
            <Condition Type="OTHERWISE"/>
   </Transition>
</Transitions>
```

Figure 7 XPDL representations of transitions

- **The mapping of decision element**

A decision element can be mapped to a route activity entity with transition restriction of spit with XOR type, which is defined in XPDL [1]. Figure 8 shows an example of XPDL representation of "Check Order Type" decision element.

```
<Activity Id="12" Name="Check Order Type">
 <Route/>
 <TransitionRestrictions>
  <TransitionRestriction>
   <Split Type="XOR">
    <TransitionRefs>
       <TransitionRef Id="24"/>
       <TransitionRef Id="25"/>
    </TransitionRefs>
   </Split>
  </TransitionRestriction>
 </TransitionRestrictions>
 <ExtendedAttributes>
  <ExtendedAttribute Name="Coordinates">
   <xyz:Coordinates xpos="293" ypos="460"/>
  </ExtendedAttribute>
 </ExtendedAttributes>
</Activity>
```

Figure 8 XPDL representation of "Check Order Type" decision element

Now we have proposed the method of the mapping from the business process model represented by extended activity diagram to XPDL document. The complete structure of XPDL document and the related detail information about associated XPDL entity definitions should be referred to in XPDL specification [1].

Conclusion and Future Work

In the paper we proposed the approach of the mapping from the business process model represented by extended activity diagram to XPDL document. However, the mapping of the element of "Swimlane" in UML activity diagram still can not be performed because of the lack of associated entities defined in the XPDL. In the future we will do more research work to solve this problem.

References

[1] Workflow Management Coalition (WfMC),2002, Workflow Process Definition Interface-- XML Process Definition Language(XPDL) specification, electronic edition

[2] Object Management Group (OMG), 2002, OMG Unified Modeling Language Specification, electronic edition.

[3] Rational, 1997, UML Notation Guide, version 1.1. electronic edition

[4] Hans-Erik Eriksson, Magnus Penker, 2000, Business Modeling with UML: Business Patterns and Business Objects, John Wiley & Sons Inc

[5] Alexander, I, A,1998, A Co-Operative Task Modeling Approach to Business Process Understanding, workshop on Object-Oriented Business Process Modeling, ECCOOP 98.