Mapping from BPMN-Formed Business Processes to XPDL Business Processes

Jung, Moon-young, Hak Soo Kim, Myung Hyun Jo
Kyung Hyun Tak, Hyun Seok Cha, Jin Hyun Son

Department of Computer Science and Engineering, Hanyang University, Korea
{myjeong, hagsoo, mhjo, khtak, hscha, jhson}@cse.hanyang.ac.kr

ABSTRACT

In the business process management, many business process execution languages such as XPDL, BPML, BPEL4WS have been specified with different origins and goals. Most of all, XPDL proposed by WfMC has been widely used in the related applications, especially workflows whose concepts are currently interchangeable with those of business processes. On the other hand, Business Process Modeling Notation (BPMN) driven by BPMI has recently been specified as a standardized graphical notation for a business process. We can therefore commonly design and analyze various business processes using the design tools to support BPMN. Notice that a BPMN-formed business process should be converted to its semantically equivalent business process languages such as XPDL which can consequently be executed by business process engines. In this regard, we propose a transformation mechanism from BPMN-formed business processes to corresponding XPDL processes.

Keywords: BPMN, XPDL, business process

1. INTRODUCTION

With an increasing number of business applications that automate business processes, many enterprises have recently devoted considerable attention to business process integration. The primary goal of business process integration is inter-operating the information flows between IT organizations that have used a number of various terms to describe how components can be connected together to build complex business processes. For instance, if our system tries to connect a partner during operating a business process, that is composed of tasks such as catalogue request, order, order processing, and dispatching, we must be required business process integration technologies. It is necessarily required many costs for reducing the difference between methodologies of each vendor. For merging business processes, which are located in distributed environment, there is a standard interface that provides flexibility or interoperability between business processes. To streamline business operations, it is important to define a standard business process language in the first place that could be adapted in complex business circumstances. It is a same reason that the computing resource is composed by the programming language such as C or Java. However, a business process language is not relied on a physical object such as operating systems or devices, but it must be designed suitably in a logical meaning of a business environment. Especially, it should be defined a mechanism that one internal partner connects dynamically the other external partners in distributed environments. In order to meet these requirements, researches of the business process language have been broadly going to two trends.

The first of these trends is a research in business process modeling language. It is lately remarkable by Business Process Modeling Notation (BPMN) Working Group. Although a business analyst doesn't know any information about an internal mechanism of a business process execution, it allows him to design a business process using a business process modeling notation. Examples of other notations or methodologies that were reviewed are UML Activity Diagram, UML EDOC Business Processes, IDEF, ebXML BPSS, Activity-Decision Flow (ADF) Diagram, RosettaNet, LOVeM, and Event-Process Chains (EPCs). BPMN has more functionality and extensibility than other business process modeling languages. So, we select BPMN for mapping from a business process modeling language to a business process execution language.

The second of these trends is a research in business process execution language. It enables a system to understand all of the information which makes our business runs. Representatively, Business Process Modeling Language (BPML) is developed by the Business Process Management Initiative (BPMI), Business Process Execution Language for Web services (BPEL4WS) is written by developers from BEA Systems, IBM, and Microsoft, and XML Process Definition Language is announced by the Workflow Management Coalition (WfMC). And they are XML-based business process execution languages. To minimize a modification of existing workflow applications using XPDL, we select XPDL among a number of business process languages.

Although there is a significant difference between two trends, both in motivating business process modeling and in their features, they have a strong common basis. Thus, the convergence of two major trends is creating a rapidly growing demand for a new breed of software that facilities automation of business processes both between enterprises and within an enterprise. However,

the difference between business processes represents a partial meaning difference, but it raises enormously the risk. In this paper, we describe how the mapping mechanism reduces a meaning difference between BPMN and XPDL. Figure 1 illustrates the mapping structure of this paper; in the future we will plan to study mapping mechanisms from BPMN-formed business processes to BPEL4WS and BPML business processes.

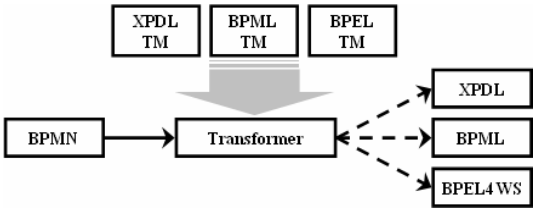

Figure 1. Transformation structure of BPMN

The remainder of the paper is organized as follows. Section 2 presents an overview of related work. Section 3 describes Mapping from BPMN to XPDL. Section 4 contains concluding remarks.

2. RELATED WORK

There are several researches in mapping from business process modeling notation to business process execution language [1,3,4,5,6]. This Section describes mapping mechanisms which have already been proposed.

[3] describes the mapping from BPMN to XPDL. It shows that BPMN notations can be transformed to XPDL straightforwardly, because of similar business process structures within them. However, there are BPMN notations which can not be mapped to XPDL, straightforwardly. Therefore mapping is applicable to only 10 BPMN notations. So, we will show the mechanism which can be applicable to other BPMN notations, either, in this paper.
[1] describes the mapping from BPMN to BPEL4WS. It is accomplished by analyzing cases which BPMN business process can be expressed. It also describes all elements and attributes of BPMN mapping to BPEL4WS elements. So, the number of pages of [1] is more than 50. It may be complete, but it is too complex to be proved.
[4] and [5] describe the mapping from UML Diagram to XPDL. [4,5] shows business process can be expressed using UML Diagram, and be mapped to XPDL. [4] describes it using Use Case Diagram, Statechart Diagram, and Activity Diagram with stereotypes, which is defined extendedly, such as <<Route>>, <<No>>, <<Tool>>, <<Subflow>>, <<Loop>>. [5] describes it using only Activity Diagram with stereotypes, which is defined extendedly, such as <<Business Document>>.
[6] describes the mapping from UML Diagram to BPEL4WS. It shows business process can be expressed using Class Diagram and Activity Diagram with stereotypes, which is defined extendedly, such as <<Process>>, <<Activity>>.

As shown in above, many researches select UML Diagrams as a notation for business process modeling more actively than BPMN. However, if we use UML Diagram to model business process, and if we want to transform the business process to a business process language, we will need to extend the stereotypes which are dependent on transformed languages.

Besides, there are several researches in expression power of business process modeling notations or business process execution languages [7,8,9,10,11,12]. (e.g., workflow patterns, Petri-net)

Like this, mapping mechanisms, which the existing papers propose, are straightforward, case by case analyzed, or system dependent formed. If a mechanism is straightforward, it will be not complete, since a difference of a notation and a language. If a mechanism is case by case analyzed, it will be too complex to be mapped, since it has to be analyzed in all cases. And if a mechanism is language dependent formed, it will be not appropriate for a standard mechanism. Therefore this paper proposes a mapping mechanism from BPMN to XPDL, which is not only simple, complete, but also system interchangeable.

3. MAPPING FROM BPMN TO XPDL

Both BPMN and XPDL are conceived of as a directed graph structure. So Mapping from BPMN to XPDL may be described straightforwardly. e.g., A Task and a Sub-Process of BPMN are transformed to an Atomic Activity and a Subflow Activity of XPDL respectively, and a Sequence Flow of BPMN is transformed to a Transition of XPDL [3]. But because BPMN is designed to model and to manage business processes and XPDL is designed to execute business processes, we need to consider some differences between BPMN and XPDL. Firstly BPMN has several elements that are not for executing business processes but only for modeling or managing business processes. These BPMN elements do not need to map to XPDL elements. Secondly, BPMN has several elements that are for executing business processes but do not transformed to XPDL elements straightforwardly. The business processes, which are represented using BPMN including these elements, have to be transformed to the same meaning XPDL business processes.

This Section describes mapping from BPMN to XPDL as follows. In section 3.1 we extract a BPMN element set, which consists of the BPMN elements that have to map to XPDL elements. In section 3.2 we look around the BPMN-formed business processes as a structural point of view and compare these with XPDL business processes. In section 3.3 we describe the mapping from BPMN to XPDL straightforwardly and in section 3.4 we

propose the transformation mechanism for BPMN elements which are not mapped to XPDL elements.

3.1 Analysis on BPMN Elements

Business processes are designed by business analysts using BPMN notations, these BPMN-formed business processes are transformed to XPDL processes for executing business processes, and the executing business processes are managed by business process managers in BPMN format. That is, the BPMN consists of elements for business process design, elements for business process execution, and elements for business process management, so several BPMN elements, which are not for business process execution, don't need to be transformed to XPDL elements.

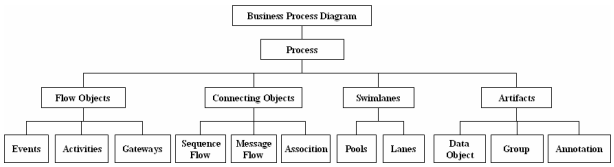

Figure 2. Major elements defined within BPMN

Figure 2 shows the major elements of BPMN and those categories. In the Business Process Diagram, the graphical objects (Flow Objects, Connecting Objects, Swimlanes, and Artifacts) define Processes (business processes); Flow Objects define the behavior of Processes, Swimlanes group other elements and assign roles to them, Artifacts provide additional information about the Processes, and Connecting Objects represent the order of Flow Objects or association between each element. But Swimlanes, Artifacts, and some of Connecting Objects (Message Flow and Association) are not related to the business process execution. First, a role, which is represented by Swimlanes, is a semantic element which means the role of doing the work (e.g. Seller, Buyer, Client, and etc.). But the element, which we need to know for executing business process, is not a role, but a performer. Second, Artifacts and Association are not directly related to the flow of process. Third, Message Flow shows the flow of messages between two entities, but the actual message is not transmitted by Message Flow but by applications or web services which is bound by Flow Objects. Therefore we don't need to transform these BPMN elements to XPDL elements. Moreover, each BPMN elements have their attributes, and some of the attributes (i.e. Name, Author, Language, CreationDate, ModificationDate, GraphicElements, Status, Categories, InputSets, Input, OutputSets, Output, BoundaryVisible) are not related to business process execution. And there is no need to transform these attributes to XPDL elements, either. Figure 3 shows the BPMN elements, except the elements which are described above. That is, these elements are related to business process execution and have to be transformed to XPDL elements.

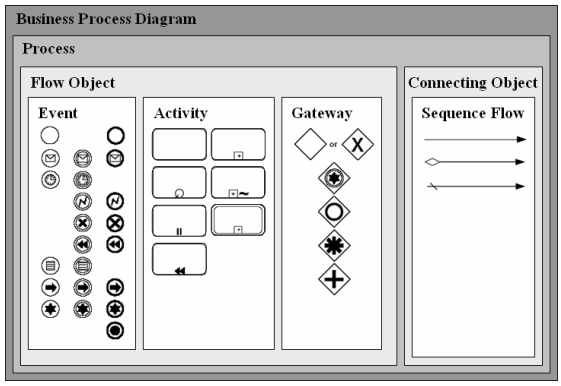

Figure 3. BPMN elements which have to be transformed to XPDL elements

3.2 Structural Mapping

A business process is a behavioral flow which is structured by performers, services, and data. In a business process, works, which is performed by performers using services and data, are defined orderly.

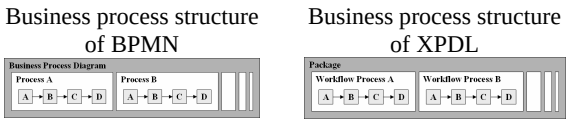

Figure 4. Business process structures of BPMN and XPDL

BPMN and XPDL represent business processes similarly, as shown in Figure 4. Processes, which are defined by BPMN elements, are structured by Flow Objects and Connecting Objects in directed graph format. Also, Workflow Processes, which are defined by XPDL, are structured by Activities and Transitions in directed graph format. Like this, Business Process Diagram, Processes, Flow Objects, and Sequence Flows in BPMN are mapped to Package, Workflow Process, Activity, and Transition in XPDL respectively.

Table 1. Structural mapping

BPMN element	XPDL element
Business Process Diagram	Package
Process	Workflow Process
Flow Object	Activity
Connecting Object	Transition

And performers, services, and data which are the elements of business processes are represented Performer, Implementation, Property in BPMN, and Performer, Tool, Data Field in XPDL.

Table 2. Business process elements mapping

BPMN element	XPDL element
Performer (within Activity)	Performer (within Activity)
Implementation (within Activity)	Tool (within Activity)
Property (within Process and Activity)	Data Field (within Workflow Process)

3.3 Simple Mapping

In Section 3.2, we described that BPMN Flow Objects and Connecting Objects are mapped to XPDL Activities and Transitions. But BPMN Flow Objects and Connecting Objects are classified with Events, Activities, Gateways, and Sequence Flows, and these are classified more detailed according to the value of type related attribute. This Section describes mapping from these detailed BPMN elements to XPDL elements which are transformed straightforwardly.

3.3.1 Events

BPMN Event acts as an event, which occurs, or reacts against another event. (e.g., cancel of an order, modification of an order, and handling these events) And it is classified according to its position (Start, Intermediate, End) and the type of Trigger (Message, Timer, Error, Cancel, Compensation, Rule, Link, Complex, Terminate).

Table 3. BPMN Event mapping to XPDL

BPMN element	XPDL element
None Start Event	Route Activity
None End Event	Route Activity
Message Start Event	Route Activity and Formal Parameters of Workflow Process
Message End Event	Atomic Activity and Route Activity
Acyclic Timer Start Event	Route Activity and No Implementation Atomic Activity including Deadline
Acyclic Timer Intermediate Event (Normal Flow)	No Implementation Atomic Activity including Deadline
Acyclic Timer Intermediate Event (Exception Flow)	Deadline of Atomic Activity

In XPDL, there are some elements which it functions as a BPMN Event. But because XPDL is on the basis of data-based control, many of them are not supported. Therefore, we describe these Events, which is not transformed to XPDL yet, in Section 3.4.

3.3.2 Activities

BPMN Activity is work that is performed within a business process. Activity is classified into Sub-Process and Task according to the subject of the work, and these are classified more detailed according to the method that it works. And we can transform it into XPDL Activity as shown in Table 4 and Table5.

Table 4. BPMN Sub-Process mapping to XPDL

BPMN element	XPDL element
Embedded Sub-Process	Block Activity
Independent Sub-Process	Subflow Activity
Reference Sub-Process	Block Activity or Subflow Activity equivalent to the Activity referenced

Table 5. BPMN Task mapping to XPDL

BPMN element	XPDL element
Service Task	Atomic Activity
Receive Task	Atomic Activity
Send Task	Atomic Activity
User Task	Atomic Activity including Performer
Script Task	Atomic Activity including Extended Attribute which can contain the script
Manual Task	Manual Mode Atomic Activity
Reference Task	Activity equivalent to the Activity referenced

We didn't describe the Mapping from Ad-Hoc Sub-Process to XPDL yet. But because it is a kind of complex mapping, we describe it in Section 3.4.

3.3.3 Gateways

BPMN Gateway controls the flow of both diverging and converging Sequence Flow. And the types of Gateway are classified according to split/merge and the function. In XPDL, Route Activity implement split or join transitions, and it works as BPMN Gateways according to Transition Restriction element within Route Activity.

Table 6. BPMN Gateway mapping to XPDL

BPMN element	XPDL element
Exclusive Decision (XOR) – Data-Based	XOR Split Route Activity
Exclusive Merge (XOR) – Data-Based	XOR Join Route Activity
Inclusive Decision (OR)	AND Split Route Activity
Inclusive Merge (OR)	AND Join Route Activity
Parallel Fork (AND)	AND Split Route Activity
Parallel Join (AND)	AND Join Route Activity
Complex Decision / Merge	Combination with several Route Activities and Transitions

We didn't describe the mapping from Event-Based Exclusive Decision to XPDL yet. But because it is a kind of complex mapping, we describe it in Section 3.4.

3.3.4 Sequence Flow

BPMN Sequence Flow shows the order of Flow Objects. And the types of Sequence Flow are classified according to the value of Condition attribute. In XPDL, as described in Section 3.3, Transition implements like this. And the type of Transition is classified according to Condition element, either.

Table 7. BPMN Sequence Flow mapping to XPDL

BPMN element	XPDL element
Sequence Flow	Transition
Conditional Sequence Flow	Transition including CONDITION Type Condition
Default Flow	Transition including OTHERWISE Type Condition

3.4 Complex Mapping

Complex mapping is the mapping for BPMN elements (complex elements) which XPDL didn't have as the same function. That is, if we recompose a business

process, which is composed of BPMN elements including the complex elements, to a business process, which is composed of BPMN elements except complex elements, we can transform the business process to XPDL. It is the same as if "A = B" and "B = C", then "A = C". This paper proposes following three mechanisms for these transitive mapping.

Mechanism 1. Loop

The Loop means that it works more than one time. We can apply this mechanism to the complex element, if the element has the semantic of "it works repeatedly" or "it works again".

Figure 5. Loop transformation mechanism

Mechanism 2. Discriminator

The Discriminator means that it completes only a work of all other works which are implemented concurrently. We can apply this mechanism to the complex element, if the element has the semantic of "it finishes B or C".

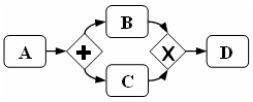

Figure 6. Discriminator transformation mechanism

Mechanism 3. Serialization

The Serialization means that it transforms something, which is serialized implicitly, to another thing serialized explicitly. We can apply this mechanism to the complex element, if the element has the semantic of "it works in the order of a basis".

Figure 7. Serialization transformation mechanism

Firstly, Cyclic Timer Event, Standard Loop Activity, Sequential MultiInstance Loop Activity, and StartQuantity attribute of Activity apply to the Loop; Cyclic Timer Event represents the work which is implemented in a cycle. Standard Loop and Sequential MultiInstance Loop Activity represents the work which is implemented repeatedly according to the Condition of Activity. StartQuantity attribute of Activity represents the work which waits until the number of token is satisfied. And the Figure 8 shows the application of the Loop transformation mechanism to these BPMN elements.

Secondly, Message Intermediate Event (Exception Flow), Error Event, Multiple Event, Terminate Event, and Event-Based Exclusive Decision apply to the Discriminator; Both Message Intermediate Event (Exception Flow) and Error Event mean that whether it flows normal flow or exception flow. Multiple Event means that if one of the Event, which is assigned to itself, occurs, it will be triggered. Terminate Event means that it interrupts all the works, which is working in the process, and end the process at once instead of normal end. Event-Based Exclusive Decision means that only one of the Events, which are in order, can be triggered. And the Figure 9 shows the application of the Discriminator transformation mechanism to these BPMN elements.

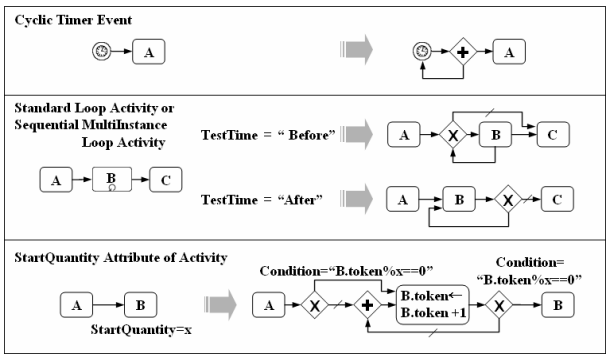

Figure 8. Loop transformation class

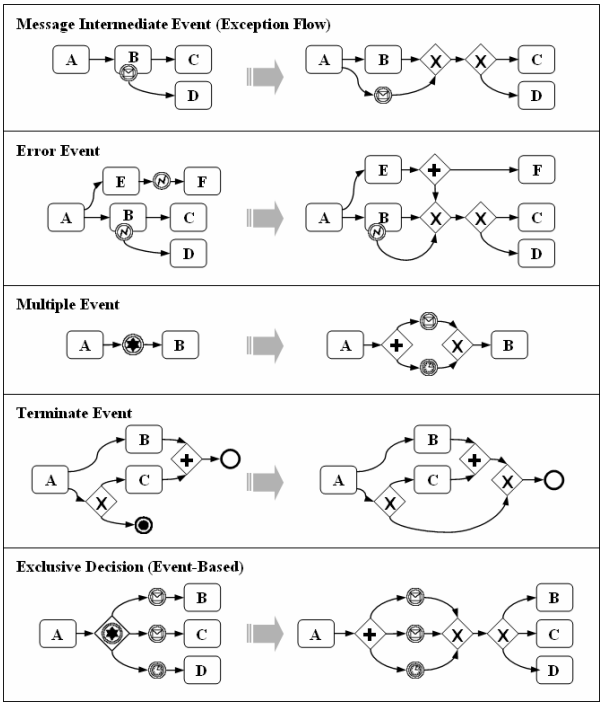

Figure 9. Discriminator transformation class

Thirdly, Link Event and Parallel MultiInstance Loop Activity apply to the Serialization; Link Event represents the work which is to be connected to another Link Event which has same LinkId. Parallel MultiInstance Loop Activity means that it works same work in parallel. And Figure 10 shows the application of the Serialization transformation mechanism to these BPMN elements.

Lastly, Ad-Hoc Sub-Process is the Process which the order of the works and the end of the process is decided by performers at execution time. So it works beginning with the work, which is selected by a performer, repeatedly until the CompletionCondition of the Ad-Hoc Process is satisfied. Therefore Ad-Hoc Process applies to both Loop and Discriminator. Figure 11 shows the application of the transformation mechanisms to Ad-Hoc Sub-Process.

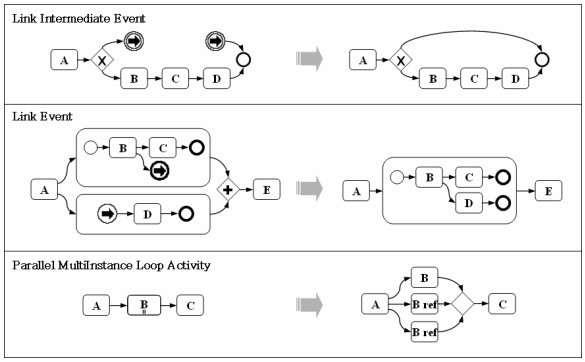

Figure 10. Serialization transformation class

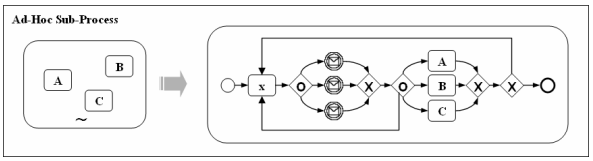

Figure 11. Transformation of Ad-Hoc Sub-Process

4. CONCLUSIONS

This paper focused on a problem which has arisen from a meaning difference between BPMN and XPDL. We have analyzed elements that are mapped from BPMN to XPDL, and then have proposed three transitive mapping mechanisms (loop, discriminator and serialization).

While the existing mapping mechanisms only selected elements that can be directly mapped, we discovered all of elements that are able to be mapped with a notation's transformation without a meaning loss. These methodologies transform a complex BPMN element into several kinds of atomic and simple BPMN elements. That is, because a reconstructed business process is composed of elements, which are adapted to mapping to XPDL, we can transform the BPMN- Formed Business Processes to XPDL Processes.

We are going to develop a module with this mapping mechanism. We will also plan to study mapping mechanisms from BPMN to other business process execution languages such as BPEL4WS and BPML.

ACKNOWLEDGEMENT

This work was supported by Korea Research Foundation Grant (KRF-2004-003-D00327).
This work was supported by grant No. R08-2003-000-10464-0 from the Basic Research Program of the Korea Science & Engineering Foundation.
This work was supported by the Ministry of Information & Communications, Korea, under the Information Technology Research Center (ITRC) Support Program.

REFERENCES

[1] BPMI.org., Business Process Modeling Notation (BPMN), Version 1.0 - May 3, 2004.
[2] WfMC., Workflow Process Definition Language -- XML Process Definition Language (XPDL), Document Number WFMC-TC-1025, Document Status - 1.0 Final Draft. October 25, 2002.
[3] Stephen A. White., "XPDL AND BPMN", Workflow Handbook 2003.
[4] P. Jiang, Q. Mair, and J. Newman., "Using UML to Design Distributed Collaboration Workflows: from UML to XPDL", *The Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*, 2003.
[5] R. Eshuis, P. Brimont, E. Dubois, B. Grégoire, and S. Ramel., "Animating ebXML Transactions with a Workflow Engine", *The Eleventh International Conference on Cooperative Information Systems (CoopIS 2003)*, Catania, Sicily, Italy, 2003.
[6] T. Gardner., "UML Modelling of Automated Business Processes with a Mapping BPEL4WS", *The 17th European Conference on Object-Oriented Programming (ECOOP)*, Germany, 2003.
[7] Workflow Patterns Home Page, *Available from http://www.workflowpatterns.com*
[8] Stephen A. White., "Process Modeling Notations and Workflow Patterns", Workflow Handbook 2004.
[9] W.M.P. van der Alast., "Patterns and XPDL: A Critical Evaluation of the XML Process Definition Language", 2002.
[10] P. Wohed, W.M.P. van der Aalst, M. Dumas, and A.H.M. ter Hofstede., "Pattern Based Analysis of BPEL4WS", QUT Technical report, FIT-TR-2002- 04, Queensland University of Technology, Brisbane, 2002.
[11] W.M.P. van der Aalst, M.Dumas, A.H.M. ter Hofstede, and P. Wohed., "Pattern Based Analysis of BPML (and WSCI)", QUT Technical report, FIT-TR-2002-04, Queensland University of Technology, Brisbane, 2002.
[12] Robert Shapiro., "A Technical Comparison of XPDL, BPML and BPEL4WS", 2002.
[13] David Hollingsworth., "The Workflow Reference Model: 10 Years On", Workflow Handbook 2004.