Apply MDA to E-Business: MDA Based Workflow Solution as an Example

Jiawei Lee, Jungsing Jwo

Department of Computer Science and Information Engineering, Tunghai University
Taichung, Taiwan, China
jwo@mail.thu.edu.tw

ABSTRACT

This paper introduces how to apply OMG's Model-Driven Architecture (MDA) to develop a cross-platform workflow solution. A workflow solution first is modeled as a PIM Platform Independent Model model. The PIM model will be translated into a PSM (Platform Specific Model) model according to the selected workflow platform and then the corresponding process definition is automatically generated. In this paper the result shows MDA can help preserve the knowledge of an application as a PIM model. The quality of the application also can be dramatically increased since the translations from PIM model to PSM model and from PSM model to the final production codes are automatically performed.

Keywords: e-Business, MDA, meta-model, WfMC, workflow management system

1. INTRODUCTION

In an enterprise workflow solution is one of the major building blocks for e-Business. In fact, workflow related solutions are widely studied recently [3, 4, 5, 7, 17, 19]. Among these researches, WfMC (Workflow Management Coalition) is the key organization to define the standard for workflow management system [4]. However, even with the standard, the migration of a workflow application from one platform to the other is still quite difficult since different workflow platforms have different philosophy to model their workflow processes.

MDA (Model-Driven Architecture) is a new technology for developing software applications proposed by OMG (Object Management Group) [12-15]. In the past, an application is modeled by also considering the IT platform and therefore the software design can not be reused when the platform is changed [1, 2, 8, 9, 18]. In MDA technology, an application is modeled only from the application logic without considering the IT platform [16]. This model is called a PIM (Platform Independent Model) model. When the IT platform for the application is decided, a corresponding model translated from the PIM model is created automatically. The new model is called a PSM (Platform Specific Model) model. Based on the PSM model and the translation rules, the production codes of the application are also generated automatically [10, 11]. Since the knowledge of the application is captured in the PIM model, it can be reused much easier later when the application needed to be migrated to another platform. The quality of the application is also dramatically increased since the translations from PIM model to PSM model and from PSM model to the final production codes are automatically performed.

In order to show the superiority of adopting MDA technology for building e-Business related solutions, we propose a new workflow application development methodology based on MDA. Figure 1.1 is the illustration of the idea. We define a workflow PIM meta-model named MyWF meta-model. In this paper, we chose WfMC's XPDL process model as the experimental PSM meta-model. We also define the translation rules for MyWF model to the XPDL model and for XPDL modelc to the final XPDL process definition.

The organization of this paper is as follows. In section 2 we briefly introduce WfMC's XPDL model. MyWF meta-model is defined in Section 3. The implementation of the translation rules is described in Section 4. Section 5 is the conclusion remark of this paper.

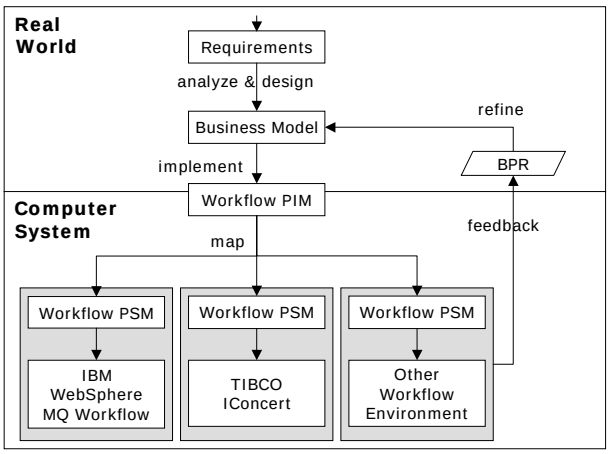

Figure 1.1. The architecture of applying MDA to build workflow application

.

2. XPDL

According to the workflow reference model proposed by WfMC, a workflow application has three different modes, namely, build-time functions, run-time control functions, and run-time interaction functions. Therefore, a workflow application can be considered as an

application with multiple operation steps such that it is routed automatically by following its corresponding process definition. The process definition standard of WfMC is called XPDL (XML Process Definition Language). XPDL is based on W3C's XML Schema [19]. Figures 2.1 and 2.2 are the meta-model and the package meta-model for XPDL, respectively.

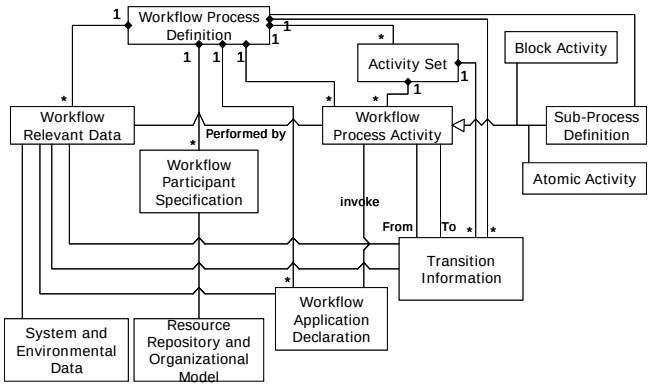

Figure 2.1. XPDL meta-model diagram.

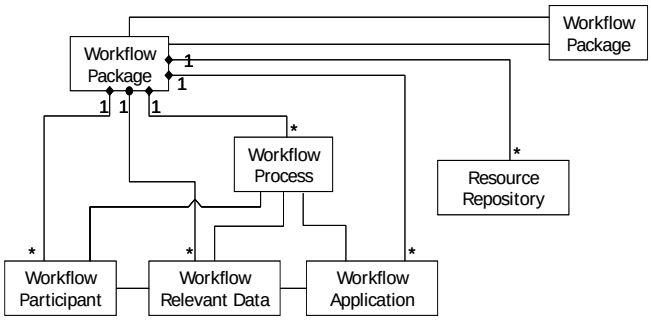

Figure 2.2. XPDL package meta-model diagram.

3. MyWF META-MODEL

In this section we define a generic workflow platform independent meta-model named MyWF. Figures 3.1and 3.2 are the design of the MyWF meta-model. In this PIM model, there are five elements, namely, Process, Task, Transition, Resource and RelevantData. Process element is used to describe a workflow's operation steps. Task element is to store the information of the operation in each step. The information about the routing between two steps is defined in Transition element. The information of the resource required for the workflow application is stored in Resource element. RelevantData element contains the information across different steps. Each Process element can have more than one Task and various Transition, Resource and RelevantData elements. Task is subdivided into two different types, namely, GroupTasks and SubFlow elements. Resource could be either Application resource or user Role elements.

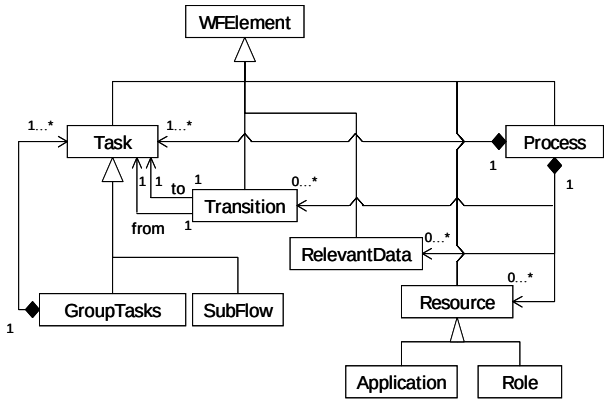

Figure3.1. MyWF meta-model diagram.

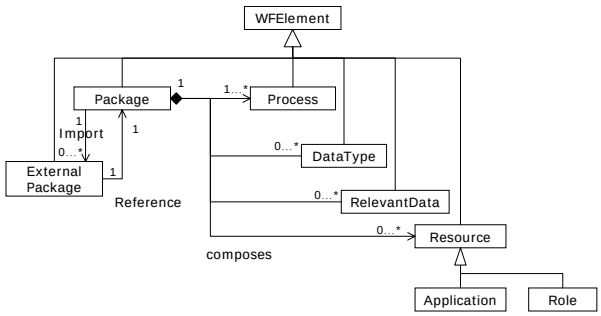

Figure 3.2. MyWF package meta-model diagram.

4. MyWF TO XPDL TO PRODUCTION CODE TRANSLATION

In the previous two sections, we have a generic workflow PIM model called MyWF meta-model and a well known workflow PSM model named XPDL meta-model. In this section, we describe how to transfer a MyWF application model into XPDL model and then transfer theXPDL model into its corresponding production codes.

4.1 MyWF PIM Model to XPDL PSM Model

The translation design between MyWF PIM model and XPDL PSM model is separated into two solutions. The first part is to guarantee that the traversal of the PIM model elements is complete. The second part is to define the translation rules among the elements between two models. We use Bean Script Framework (BSF) to implement the two solutions. The reasons of selecting BSF to build the MDA translation solution are:

- *Flexibility*: the interpreting based script language is more flexible when translation rules are changed and are required to be re-implemented.
- *Friendly implementation*: BSF is easier for MDA developers to use when implementing the translation rules among different models.

The translation algorithm is described as follows.

(1). Retrieve the next element in the MyWF PIM model.
(2). Identify the element type of the current element. If the type is unknown then throw UnknownTypeException and interrupt the traversal.
(3). According to the translation rules, fetch and execute the corresponding script code and perform the translation. If the script is not found, throw NoScriptException and interrupt the translation step.
(4). Retrieve the related translation required information and put it into the ScriptEngine.
(5). Execute the corresponding script code to build the translated XPDL elements. If the execution can not be executed correctly, throw TransfomrationException and interrupt the execution.
(6). Inspect the current MyWF element and make sure if there is any related element. If yes, go to step 1, otherwise go to step 7.
(7). Return the translated XPDL elements. Go to step 1.

4.2 WfMC XPDL to Production Codes

In this section, we describe the translation rules between PSM model and the final production codes.

For each element in the PSM meta-model, we define one or more than one templates. Each template is a code pattern. The translation algorithm is as follows.

(1). Retrieve the next element in the PSM model.
(2). Identify the element type of the current element. If it is not known, throw UnknownTypeException and interrupt the execution.
(3). Retrieve the corresponding templates. If there is no template is available, throw NoTemplateException and interrupt the execution.
(4). Parsing the template pattern code. Identify the required information for the template from the element. If there is no corresponding information, throw NoTemplateAttributesException and interrupt the execution.
(5). Return the generated production codes. Go to step

5. CONCLUSION

Figures 5.1 is the diagram to show how to define the roles and development process when applying MDA technology to build a workflow application. Figure 5.2 is the screenshot of our example implementation using Java and JMI [6]. From our experience of this research, we find that MDA indeed can greatly reduce the effort when implementing e-Business solutions on different platforms.

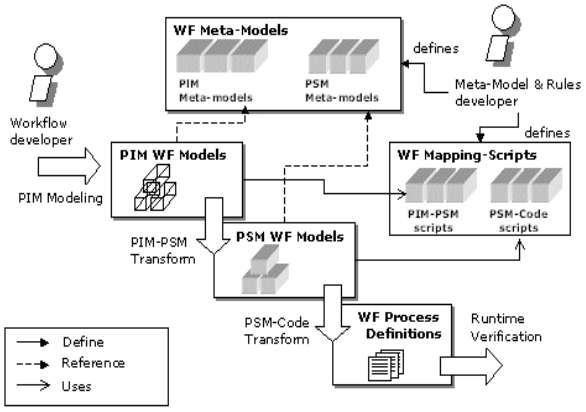

Figure 5.1. The roles and development process when applying MDA to workflow application.

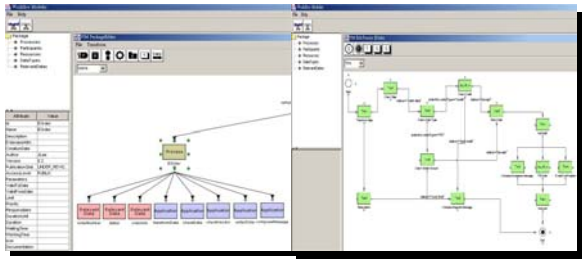

Figure 5.2. A screenshot of the example solution.

REFERENCES

[1] Ahmed Abd-Allah, “Composing Heterogeneous Software Architectures,” Doctoral Dissertation, Center for Software Engineering, University of Southern California, Los Angeles, CA 90089, August 1996.
[2] Alessandro Bianchi, Danilo Caivano, Giuseppe Visaggio, “Iterative Reengineering of Legacy Systems,” IEEE Transactions on Software Engineering, volume 29, issue 3, March 2003, pp. 225-241
[3] Albano, F.; Pino, J.A.; Borges, M.R.S., "Participatory business process reengineering design: generating solutions," IEEE SCCC 2001, pp. 13 –22
[4] David Hollingsworth, “Workflow Management Coalition The Workflow Reference Model,” Document Number WfMC-TC-1003, 19-Jan-95, 1.1, http://www.WfMC.org
[5] Diimitrios Georgakopoulos, Mark Hornick, Amit Sheth, “An Overview of Workflow Management: Form Process Modeling to Workflow Automation Infrastructure,” Distributed and Paralled Databases, pp. 119-153
[6] JMI Expert Group, “Java Metadata Interface (JMI) Specification,” Version 1.0, 07-June-2002, http://java.sun.com/products/jmi/
[7] Keith D. Swenson, Kent Irwin, "Workflow technology: trade-offs for business process re-engineering," Conference on Supporting Group Work, 1995, pp. 22-29
[8] L. Davis, R. Gamble, and J. Payton, ”The Impact of Component Architectures on Interoperability,” Journal

of Systems and Software, Volume 61, Issue 1, March 2002, pp. 31-45.
[9] Ling Feng, Hongjun Lu, Allan Wong, "Integrating Legacy Systems Towards Intelligent Enterprise Computing," IEEE SMC '99 Conference Proceedings, Volume 2, 1999, pp. 184-189.
[10] Milicev, D.; "Automatic Model Transformations Using Extended UML Object Diagrams in Modeling Environment," IEEE Transactions on Software Engineering, volume 28, issue 4, April 2002, pp. 413-431.
[11] Oldevik, J.; Solberg, A.; Elvesæter, B.; Berre, A.J.; "Framework for model transformation and code generation," IEEE Enterprise Distributed Object Computing Conference, 2002, pp. 181-189.
[12] OMG, "Model Driven Architecture (MDA)," OMG Document number ormsc/2001-07-01, http://www.omg.org
[13] OMG, "Meta Object Facility Specification," OMG Document number formal/02-04-03, http://www.omg.org
[14] OMG, "XML Metadata Interchange Specification," OMG Document number formal/2002-01-01, http://www.omg.org
[15] OMG, "Unified Modeling Language Specification," OMG Doucment number formal/2003-03-01, http://www.omg.org
[16] Snoeck, M.; Dedene, G.; "Experiences with Object Oriented Model-Driven Development," IEEE Software Technology and Engineering Practice Proceedings, 1997, pp. 143-153.
[17] Schmidt M.T. ,"The evolution of workflow standards," IEEE Concurrency, Volume: 7 Issue: 3, July-Sept. 1999, pp. 44-52
[18] Sakib Abdul Mondal, Kingshuk Das Gupta, "Choosing a middleware for web-integration of a legacy application," ACM SIGSOFT Software Engineering Notes, Volume 25, Issue 3, May 2000, pp. 50-53
[19] Workflow Management Coalition, "XML Process Definition Language Specification (Draft 1.0)," Document Number WfMC-TC-1025, http://www.WfMC.org