BPIDE: A Business Process IDE Supporting Multi-Role Collaboration

Fanxin Deng, Hailong Sun, Xiang Li
School of Computer Science and Engineering
Beihang University
Beijing 100191, China
{dengfx, sunhl, lixiang}@act.buaa.edu.cn

***Abstract*—Business Process Management aims at creating business values for enterprises in a more flexible and competitive way. However, an integrated development environment for developing business processes with good support of multi-role collaboration is still rarely studied. In this paper, a business process integrated development environment called BPIDE based on a role-sensitive framework is designated to provide different roles (especially the analysts and developers) in different stages to work on business processes together in a more efficient way. We also elaborate on the implementation of BPIDE using a new development pattern. At last, a case study is used to demonstrate how different roles can develop business processes collaboratively with BPIDE.**

Keywords - Business process; BPM; IDE; BPMN; service composition;

I. INTRODUCTION

In recent years, BPM (Business Process Management) [1] has become the mainstream solution to creating business values for enterprises in a more flexible and competitive way. With BPM, business processes are treated as the central software artifacts that are eventually enacted by process aware information systems. The activities constituting BPM include design, modeling, execution, monitoring and optimization [2]. In this paper, we focus on the development phase of business processes.

There are several roles participating in the BPM lifecycle, like analysts, developers, testers and managers. Generally speaking, two of them play the most important part in developing business processes, i.e., analysts and developers. Among them, analysts are responsible for analyzing the business requirements, designing and modeling high-level business processes. On the other hand, developers are concerned with how to make processes executable. Additionally, other roles need to involve in other development activities, such as verification, validation and test. Therefore, the efficient collaboration between different roles and the corresponding activities is a very important issue in the business process development. Approaches are needed to maximize the development productivity when different roles are working on business processes. A developing environment is needed to reduce the development cycle and communication costs between different roles.

Traditionally, a lot of tools, such Visio BPMN Modeler [4], are designed for analysts to analyze and model business processes in a direct and natural way using standards like EPC [5] and BPMN (Business Process Modeling Notation) [6]. However, these tools are much like drawing tools. The process artifacts produced by these tools are often used by other roles like developers in a human-understandable way. So developers will make efforts on implementation according to these business processes represented in a graphical form or documents. There are several IDEs available to enact business processes. For example, developers can use Eclipse [7] to develop systems through programming in Java to realize business processes. They can also use the Synthy IDE [8] to compose web services to execute business processes.

However, due to the continuous communication for ensuring that different roles have the same understanding of business processes, this will incur much cost for transmitting business processes from analysts to developers. And this problem is much more serious considering on-demand design and development.

In this paper, we first identify and characterize the problem of multi-role collaboration. Second, a solution is proposed based on BPMN and the reason of adopting BPMN is given. Third, we demonstrate the design of an IDE called BPIDE supporting multi-role collaboration based on the solution. Finally, we present our experience of implementing BPIDE. The implementation approach provides the flexibility in terms of further improving the development of BPIDE and considers the change of the process meta-model due to the change of process modeling notations [1], work flow patterns [9], choreography support, and so on. Our main contributions are as follows:

- A characterization of the problem of multi-role collaboration and a solution based on uniform model by extending BPMN.
- An IDE supporting multi-role collaboration across full BPM life-cycle that integrated various tools based on the solution.
- A Modeling + Coding Development Pattern that improves the extensibility and flexibility of the IDE.

The structure of the rest of the paper is as follows: in Section 2, we analyze the challenging issues behind multi-role collaboration. Section 3 presents the design of BPIDE supporting multiple roles collaboration. We show the implementation of BPIDE in Section 4. Section 5 gives a case study. In Section 6, related work is discussed. We conclude this work in Section 7.

978-0-7695-3862-4/09 $26.00 © 2009 IEEE
DOI 10.1109/ComputationWorld.2009.44

II. MULTI-ROLE COLLABORATION IN BUSINESS PROCESS DEVELOPMENT

It is not enough to just model a business process. Besides that, we must consider to realize the behaviors using process artifacts with enough execution information like using BPEL4WS [10]. A process integrated development environment is an environment supporting developing these processes effectively. Usually, in several stages of developing processes there are several roles. Every role is often responsible for one certain aspect of business processes, so different roles concern on different process elements, namely, every role has one perspective. For example, for developers the process elements they concern on are the elements expressing implementation information like web service, environment variables and other technologies. This leads to the requirements of the IDE. Like shown in Figure 1, different roles usually need different functions and user interface, with which they can contribute to the development of processes.

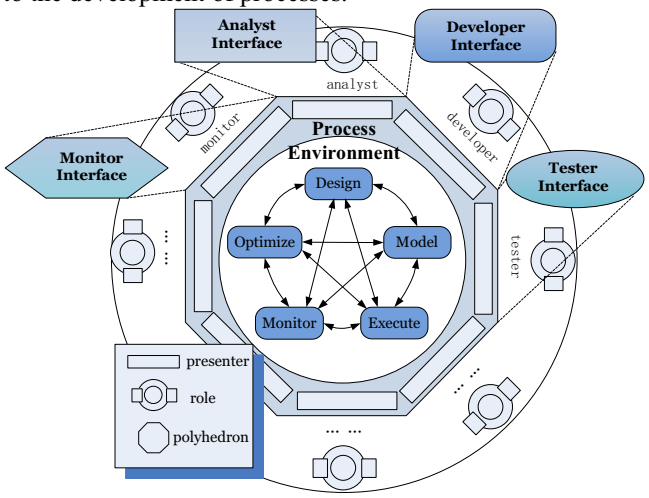


Figure 1. Overview of multi-role collaboration in business process development

The state of processes may transits from one state to another state when one role finish an activity on processes and another role begins another activity. During the state transition, the IDE is required to help complete the smooth handover from one role to another role. For example, when analysts need to change processes, the environment will transmit the processes that developers just modified to analysts while it must make sure the process elements that developers add or delete will not influence the process elements that analysts create. As best, we hope such transition can be completed automatically by process development environment. The octahedron in Figure 1 indicates a polyhedron with n faces means that the process environment can present different aspects of processes according to different roles and meanwhile the real carriers of processes often does not expose themselves.

The multi-role collaboration problem can be concluded as followed: All involving roles will act on a set of processes which will be managed by the process environment while for different roles different user interfaces will be presented and the transition between different roles should satisfy: the change of business processes one role has made should be reflected in other roles' perspective as soon as possible.

Next, we give the overview of the solution.

A. *Uniform Model of Business Processes*

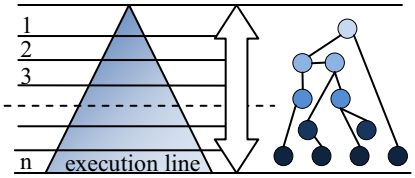


Figure 2. Process refinement model

In the context of multi-role collaboration, the development of process can be seen as continuous refinement of business processes (Figure 2). From the start, analysts can analyze, create business processes. Then they can continuously refine the processes until in the analyst's viewpoint the processes satisfy the business requirements. After that, developers will continue refining processes by filling technical information into business processes until the processes can be executed by runtime infrastructure. In traditional development of processes, the two main stages often involve different process artifact type (i.e. process representation type). For example, analysts often use a BPMN Diagram to express process information in a graphical way. Developers often use BPEL files to express more detailed process information. This leads to the difficulty of the transition from one level to another level, because the process elements in different process representation types are often different and the transformation is often hard. One solution is to define the formal transformation between different process artifact types so that the transformation is transparent to users. Another solution is using the uniform model of business processes through all stages. This means that for different roles the process artifact type is the same. The latter means gets out from the transformation trouble, and provides a smooth and round-tripping transition between different roles.

The uniform model solution is adopted by our IDE to support multi-role collaboration. In this solution, the only difference between different roles is that they will maintain different process elements. Like the nodes indicating process elements and links indicating relationships between elements shown in Figure 2, at high level some elements will created by high level role and according lower level role will add new elements containing more detailed information and create relationships with higher level process elements. This procedure will continue until business processes can be enacted into the runtime infrastructure. Once the elements at high level elements are removed, the related relationships will be removed. So, every role is responsible for a certain element type and relationships will maintained by process environment.

In the later of this paper, we will describe how we extend BPMN to realize this solution.

B. Role-Sensitive Framework

To make the IDE be able to present different user interfaces and functions like verifications to different roles, a framework is presented to make the whole environment role-sensitive. For example, the analysts and developers have different viewpoints. An analyst may concern more about abstract process than implementation process and will get help from human resource management tools, while the developer need the syntax check so that the process can run on engine. So the framework should be sensitive to roles and then has a center controlling the IDE.

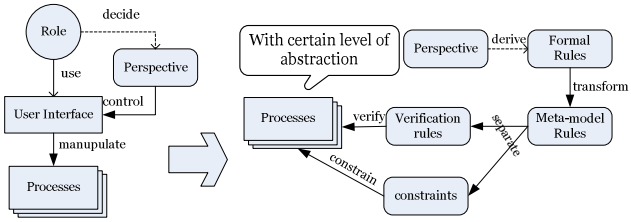


Figure 3. Role-Sensitive framework and derived verification framework

Figure 3 shows the role-sensitive framework and derived verification framework. The verification problem is a sub-problem of the role-sensitive problem. For different roles, the need to verify process model is usually different, so we need a role-sensitive verification framework which is derived from the role-sensitive framework. Other sub-problems also adopt the same meaning. All frameworks will be implemented in the IDE.

III. DESIGN OF BPIDE

In traditional software development, an integrated developing environment (IDE) is designed to maximize programmer productivity by providing tightly-knit components with similar user interfaces. This should mean that the programmer has much less mode switching to do than when using discrete development programs. In the business process IDE supporting multi-role collaboration, the basic activity is developing business processes and users of different roles contribute to the development of processes in a smooth way.

In this section, the IDE called BPIDE that supports multi-role collaboration is proposed. We will state the implementation of our solution based on BPMN and the architecture of BPIDE.

A. Process Meta Model

First, we will explain why we extend BPMN to present process information as the uniform model of business processes in our solution, then using metamodel we will declare what kinds of process elements can exist in an extended BPMN model.

1) BPMN

The Business Process Modeling Notation (BPMN) is a standard for business process modeling, and provides a graphical notation for specifying business processes in a Business Process Diagram (BPD). The primary goal of BPMN is to provide a standard notation that is readily understandable by all business stakeholders. We can hardly find another business process representation more suitable to achieve our goal, because it not only has strong expressive power, but also is graphical and analysts-friendly. Thus, BPMN provides most widely accepted way that analysts like, and analysts can use process elements in BPMN to express business processes in a natural way. Then we can extend BPMN by adding more process elements to satisfy low level business process requirements. However, the weakness of BPMN now is the lack of persistence format and formal semantics. If it is used in the implementation phase, it need formal semantics and allow for filling execution information. So based on former works on BPMN execution [11] and formal semantics [19][12][20][13], we defined the metamodel of the business process in our solution.

2) Metamodel

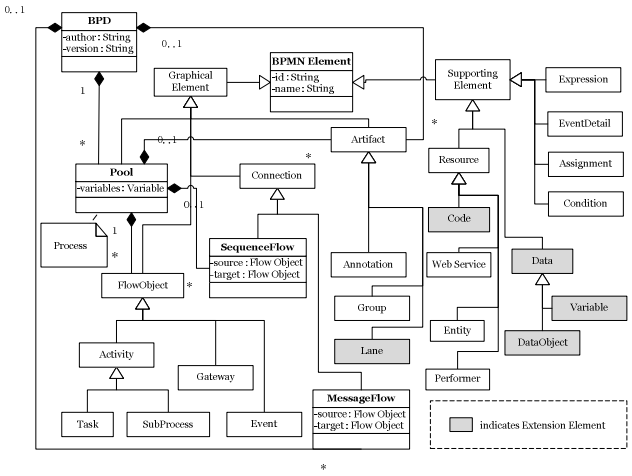


Figure 4. Main Element Types of the Metamodel

Figure 2 shows the main extended BPMN elements. From the metamodel, a Pool is treated as a Process which is the core component representing an organization of process elements. A Business Process Diagram (BPD) can contain some interacting Pools together. Message flows are used to connect either Pools themselves or Flow Objects within the Pools. Artifacts can appear anywhere as annotations having no influence on execution. In the metamodel, Lane is treated as an Artifact because it only works like Artifact to divide Flow Object into logical blocks, so Pool need not include at least one Lane anymore in comparison to that specified in BPMN Specification. A Process is called a Pool when it appears in a BPD, and is called Sub-Process when it appears in a Pool or Sub-Process. A Reference Sub-Process can be used to reference a Process.

Considering implementation technologies, firstly we allow developers define and use Variables, Expressions, Conditions, Assignments and other native functions in a Process. Secondly, implementation technologies like Web Service, programming languages are introduced into process to help perform some actions on variables that specified service providers provide. Developers can code to get the needed functions by different languages.

Now we explain how to use the same metamodel to present different form for different roles. Take the need of analysts and developers as an example to describe our mapping policy. We first divide BPMN model include two kinds of information, i.e., notations information and supporting elements information. Every notation in the model can be mapped to a graphical Node in a diagram, because according to BPMN Specification a notation has a standard node style. Further, we category supporting elements into two types: visible element for analysts, visible element for developers. A visible element for analysts is often resource such as performer and document that analysts will concern about, but it has no corresponding notation, so it is will be presented to the user in some form like property of some notation or a customized notation shown when need; A visible element for developers is often data related element which is only manipulated by developers so that the information for execution can be filled, and this kind of elements related to technique are not seen by analysts. Figure 3 shows the state that three kinds of elements exist together. Thus, by mapping notations in model into BPD, we can present a diagram to analysts while by mapping the notations and visible elements for developers into some program elements, we can present developers a language file. In other situations, we will also adopt this kind of mapping policy.

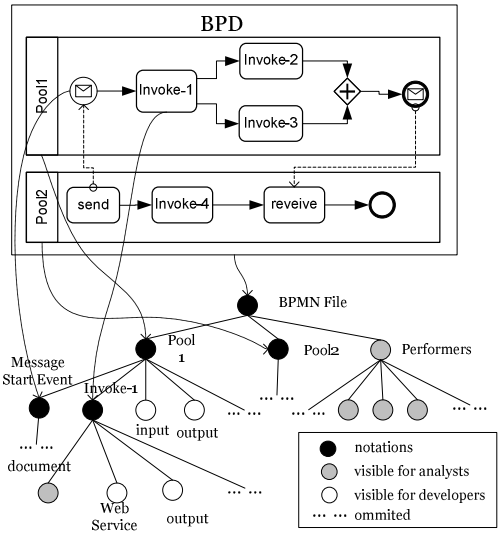


Figure 5. Mapping of Diagram and Model

Thus, in the same BPMN model, elements can be used to present different form. The BPMN model is treated as a process database, and different views for different roles are created to present different process data.

For persistence of BPMN model, based on the metamodel, a XML Schema is used to specify the representation format of the process. The full schema is available at [15] where full metamodel is also available.

B. *Architecture of BPIDE*

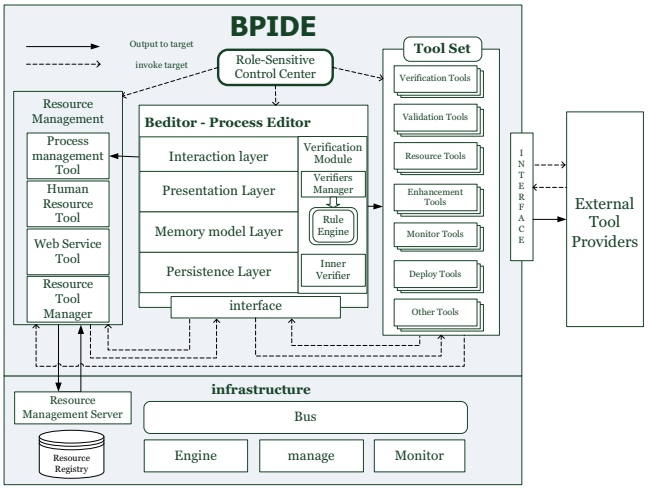


Figure 6. Architecture of BPIDE

Now we describe the design of the IDE. The architecture of BPIDE is seen in Figure 6.

The main three parts of BPIDE are Process Editor called Beditor, Resource Management Module and Toolset. All the three parts are role-sensitive and will be controlled by Role-Sensitive Control Center which will first feel the existence of roles and use corresponding perspective to control all other modules. Note that the detailed discussion of runtime infrastructure is out of scope of this paper. Now we introduce the three main parts of BPIDE.

a) Beditor

Beditor is the core of BPIDE, which is designed to facilitate development of processes in a multiple way to create business processes with efforts for the usability and extensibility of IDE. With Beditor, users of different roles can create notations or variables with the help of feedback of the interaction layer to help them perform action quickly, provide hints or prevent users from wrong actions. Presentation layer integrates modules presenting different forms to different roles. Its role-sensitive module will be controlled by role-sensitive control center. When the analysts are developing business processes, it can display all notations on the canvas in a standard way as the BPMN Specification has specified. For developers, Beditor will provide functions to edit execution information like creating variables and binding Web Services. Also for other roles, different interface will be adjusted on demand. The memory layer is responsible for manipulating the process model. It manages all information about all business process elements, and is the core part of Beditor. The memory layer also provides an interface for other tools to access process elements and manipulate process. At the persistence layer, process will be stored into files that will be treated as resources being managed by resource management tools which will facilitate reuse of resources.

At any time when a user edits process with Beditor, verification module works to provide rules to different layers. According to the verification framework proposed in Section

2, any verification tool should define its rules and register a verifier with certain rules for some goal to the verifier manager, and verifier manager will manage all verifiers and their rules. Some rules are initially embedded in Beditor by the inner verifier, because these rules are the basic rules that a BPMN diagram must follow. Every layer will get the assistance provided by the verification module. For example, in the interaction layer, rules will imply many constrains (e.g., a sequence flow cannot be linked across different pools) with which user cannot create a notation in some context, while in the memory model layer assistants like quick-fix can be performed at any time, and the persistence layer is responsible for synchronizing process model and files.

b) Resource Management

Besides managing process files mentioned above, resource management plays the important role in managing Web Service resources, human resources, data resources and other resources. Resource Management makes the best reuse of all kinds of resources by interaction with Resource Management Server and help users to create processes more effectively. In BPIDE, a number of tools are developed to manage resources under the support of Resource management server. Resource tool manager allows external resource tools to work with Beditor more effectively. The interoperation and Drag & Drop support between Beditor and resource tools will provide a good user experience.

c) Toolset

In BPIDE, a lot of tools are made to help users to accomplish tasks in different stages of developing processes. The Beditor provides the interface for them to access process model. Some of the tools are only for one role, while others are for many roles. The one-role tools are shown or hidden by BPIDE according to current perspective, while the multi-role tools will provide role-sensitive interface to be controlled by role-sensitive control center. Thus, the combination of tools and beditor can present an appropriate user interface to current role.

IV. IMPLEMENTATION EXPERIENCE OF BPIDE

For challenging the implementation of the architecture proposed in Section 4, BPIDE is built using Modeling Development Tools (MDT), Java Development Tools (JDT) in Plug-in Development Environment (PDE) on the Eclipse platform [7].

A. Eclipse Modeling Development Tools

Eclipse Modeling Tools that promote model-based development technologies within the Eclipse community by providing a unified set of modeling frameworks, tooling, and standards implementations. Modeling tools support abstract or concrete syntax development, model to text or code generation, and model transformation. Eclipse framework supports extensibility by providing extension points defined by providers where other plug-ins can add new functionality. GMF (Graphical Modeling Framework) and EMF (Eclipse Modeling Framework) are core tools of MDT. GMF is based on graphical eclipse framework (GEF) and EMF to provide a well-developed runtime infrastructure for building graphical modeling tools on it. EMF is a java framework with code generation for building tools based on structured model, and provide extension points for developers. GEF is designed to build a MVC framework that allows developers to take an existing application model and quickly create a rich graphical editor.

B. JDT & PDE

The JDT (Java Development Tools) provides the tool plug-ins that implements a Java IDE supporting the development of any Java application. The Plug-in Development Environment (PDE) provides tools to create, develop, test, debug, build and deploy Eclipse plug-ins, fragments, features, update sites and RCP products.

C. Modeling + Coding Development Pattern

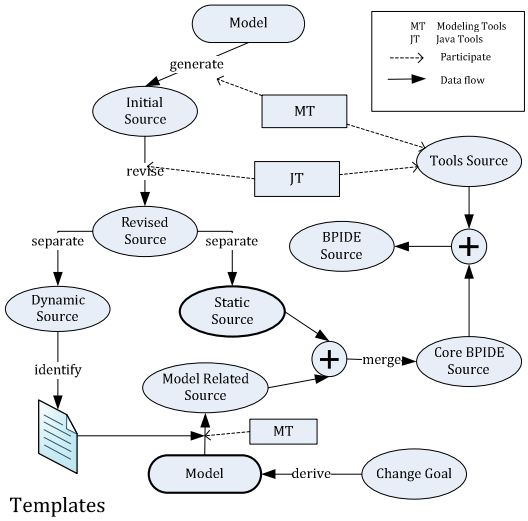
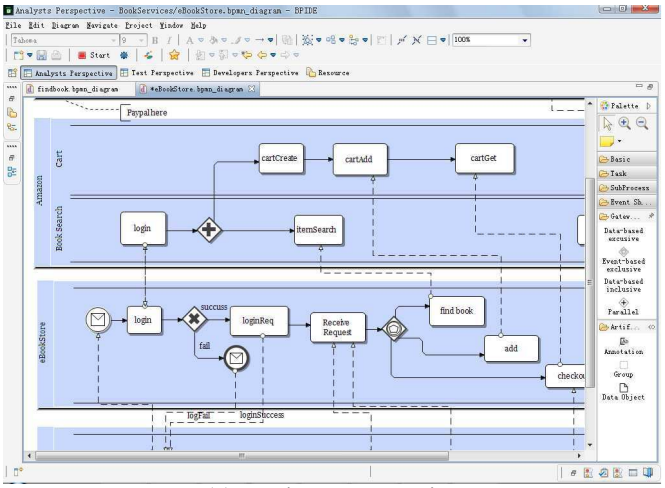
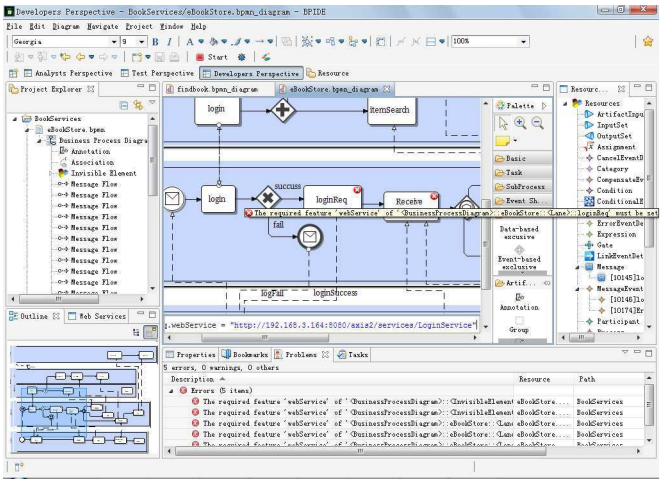


Figure 7. Modeling + Coding Development Pattern

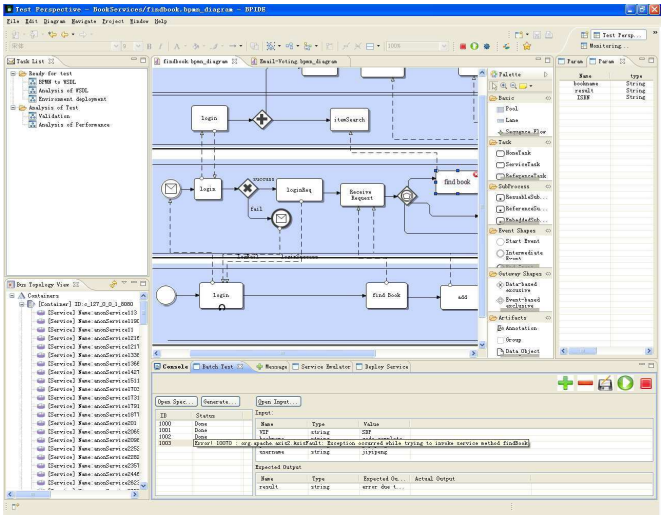
The overview of the process of developing BPIDE is shown in Figure 7. Since the development of an IDE is complex, we made efforts to reuse the existing open source resources to form the basic framework of BPIDE. First, modeling and code generation approach (GMF tool) was used to generate the initial source of BPIDE, and then by coding in Java, we revised the source according to requirements, then the new code generation templates in the form of Xpand suitable for BPIDE are identified by reconstruction and separating dynamic source and static source, so that it is agile for further metamodel change. Change of metamodel will quickly be turned into the generated model related source. After metamodel is created, static source and model related source will be merged into the core BPIDE source. The main part of the core BPIDE source is Beditor source which relies on metamodel strongly. Some of other tools are also developed using modeling tools and Java tools and then be integrated into BPIDE. Further,



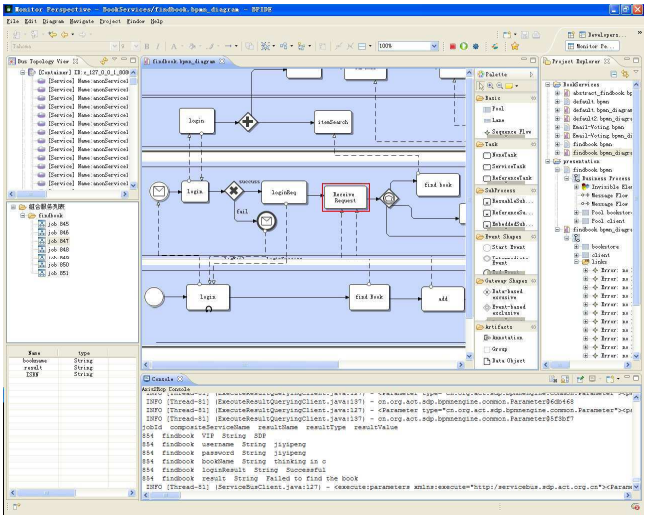
(a) Analysts Perspective



(b) Developers Perspective



(c) Test Perspective



(d) Monitor Perspective

Figure 8. Screendhots of Four Perspectives

we will provide development platform to facilitate developing of tools.

We named this process of developing BPIDE "Modeling + Coding Development Pattern" which reflects the efforts to make BPIDE more flexibility. For example, if choreography support is added into the business process modeling, some new process elements will added into the metamodel. To lower the cost of further developing BPIDE when the metamodel changes, developers of BPIDE can start from the nodes with bold dashed border in Figure 7.

The implementations of almost all tools used PDE. They can be designed to be plugged into eclipse platform as Views which provided by Eclipse SDK. In PDE we also created perspectives to make IDE role-sensitive.

V. Case Study

In this section, we will show a typical process of developing processes with BPIDE using a case inspired by [17] for creating a new service called eBook Store depending on the Amazon E-Commerce Services and an e-payment service offered by an important Italian banking Group. In our case, we changed the e-payment service to the PayPal services. Screenshots of BPIDE in four stages are shown in Figure 8 while the BPMN diagram of this case is available at [15].

First, a graphical BPMN diagram is modeled in the Analysts Perspective (see Figure 8(a)). Customers, eBook store that will be created, the Amazon Cart and Book Search Services, and PayPal Services can be put together in a diagram, and the interaction between them can be expressed by the messages interchanged between the pools.

After analysts complete designing in a graphical way, developers refine the BPMN model that should be filled more execution information so that it can run on engine. In developing, developers may assign a web service to a task using a simple SHELL command and create variables in the Resource View. Figure 8(b) shows the situation a developer encountered. The inner verifier can also allow performing verifications and showing errors. In this stage, developers can attach execution information to elements created by

analysts. Round-tripping is also available. Once analysts change the diagram, developer need only concern new elements. Feedback from developers, even testers and monitors can be easily shown on the diagram.

After development, the developed process can be deployed onto engine to test the process. In Test Perspective (see Figure 8(c)), a set of test tools based on a BPMN test approach [14] can help testers to test processes remotely. After testing the detailed result can be shown in Beditor and other Views.

Finally, once process is validated and will be published to the server. In Monitor Perspective (Figure 8(d)), from Instances View, current running process instance can be listed. By clicking the item, a corresponding process diagram file can be opened and its running state is shown in the View Mode of Beditor.

VI. RELATED WORK

The current main solution to multi-role collaboration problem is in BPMN- BPEL perspective. Typical works like IBM WebSphere [3] provide analysts with a BPMN modeler and developers with a BPEL editor. For automatic transformation between BPMN and BPEL to reduce the collaboration cost, several approaches like [21] have been proposed to transform BPMN diagram into BPEL code. The proposed techniques are capable of generating readable BPEL code by discovering "patterns" in the BPMN models that can be mapped onto BPEL block-structured constructs or acyclic graphs of control links. Now BPMN-BPEL way has been supported by most commercial tools, but it turns out to be rather complex [22] to define the transformations because BPMN diagram is graph-oriented while BPEL is block structured. This also leads to no round-tripping. If restriction is not imposed on BPMN, bidirectional mapping of BPMN and BPEL is rather hard, especially automatic round-tripping. Also high level roles cannot monitor process on their graphical diagram. Multi role collaboration still faces trouble in BPMN-BPEL solution.

VII. CONCLUSION AND FUTURE WORK

In this paper, we presented the design and development of an Integrated Development Environment (IDE) for business process development towards multi-role collaboration. The IDE is based on a BPMN-based process uniform model and a role-sensitive framework. While developing the IDE, we used a development pattern to make BPIDE development agile for future improvement. Finally, we show a case study to demonstrate how different roles can develop business processes together in BPIDE.

In future, we plan to develop other tools such as service recommendation tool, Script Support, and integrate those tools with current IDE so as to provide better support for business process development.

ACKNOWLEDGMENT

This work was partly supported by China 863 program under Grant No. 2007AA010301. And we thank Xiaotian Xu and Yang Zhao for their contribution to the development of BPIDE.

REFERENCES

[1] W. V. D. Aalst, A. H. M. Hofstede, and M. Weske. Business process management: A survey. Springer Lecture Notes in Computer Science, 2003.

[2] Wikipedia. Business Process Management. http://en.wikipedia.org/wiki/Business_Process_Management

[3] IBM. WebSphere. http://www.ibm.com/websphere [accessed, June 30, 2009].

[4] ITP-Commerce. Business Process Modeling mit Process Modeler for Microsoft Visio. http://www.itp-commerce.com/ [accessed, June 30, 2009].

[5] Y. G. Kim. Process modeling for BPR: event-process chain approach, Proceedings of the16th International Conference on Information Systems, pp.109-21,Amsterdam,2005.

[6] OMG. Business Process Modeling Notation Specification. http://www.bpmn.org/Documents/BPMN 1-2 Specification.pdf [accessed, June 30, 2009].

[7] E. Foundation. Eclipse Platform. http://www.eclipse.org/ [accessed, June 30, 2009].

[8] G. Chafle, G. Das, K. Dasgupta, A. Kumar and S. Mittal. An Integrated Development Environment for Web Service Composition. IEEE International Conference on Web Services, 2007.

[9] P. Wohed, WMP van der Aalst, M. Dumas, AHM ter Hofstede, and N. Russell. Pattern-based Analysis of BPMN-An extensive evaluation of the Control-flow, the Data and the Resource perspectives. BPM Center Report BPM-05-26, BPMcenter. org, 2005.

[10] A. Arkin, S. Askary, B. Bloch. Web Services Business Process Execution Language Version 2.0. Working Draft. WS-BPEL TC OASIS, May 2005.

[11] T. Schreiter: xBPMN++: Towards Executability of BPMN: Data Perspective and Process Instantiation. Master thesis, Hasso-Plattner-Institute, Potsdam, Germany, 2007.

[12] W.M.P. van der Aalst, A.H.M. Hofstede, and M. Weske: Formal Semantics and Analysis of BPMN Process Models using Petri Nets. Lecture Notes in Computer Science, Springer, 2003.

[13] D. Prandi, P. Quaglia, and N. Zannone: Formal Analysis of BPMN Via a Translation into COWS. LNCS 5052, pp. 249-263, 2008.

[14] R. F. Jin, H. L. Sun, X. Li, and C. Zhou. SOArTester: An Automatic Composite Service Testing System on Test Case Reduction. Accepted by National Software Application Conference, China, 2009.

[15] http://dengfanxin.appspot.com/BPIDE

[16] F. Kamoun. A roadmap towards the convergence of business process management and service oriented architecture. Ubiquity, 8(14), 2007. ACM Press.

[17] A. Marconi, M. Pistore, P. Poccianti, and P. Traverso. Automated Web Service Composition at Work: the Amazon/MPS Case Study. IEEE International Conference on Web Services, 2007.

[18] J. C. Grundy, and J. G. Hosking. Serendipity: integrated environment support for process modelling, enactment and work coordination. Automated Software Engineering of Springer, 1998.

[19] H. Endert, B. Hirsch, T. Kuster, and S. Albayrak. Towards a Mapping from BPMN to Agents. Lecture Notes in Computer Science of Springer, 2007.

[20] P.Y.H. Wong and J. Gibbons. A process semantics for BPMN.Lecture Notes in Computer Science of Springer, 2008.

[21] C. Ouyang, M. Dumas, A.H.M. ter Hofstede, and W.M.P. van der Aalst: From BPMN Process Models to BPEL Web Services. IEEE Conference on Web Service, 2006.

[22] J. Recker and J. Mendling. On the Translation between BPMN and BPEL: Conceptual Mismatch between Process Modeling Languages. In: Proceedings of the 11th EMMSAD, 2006.