ICS2002: Workshop on Database and Software Engineering

An Approach to Deploying a Process Model Quickly

Feng-Yu Chung and Feng-Jian Wang
Department of Computer Science and Information Engineering
National Chiao Tung University
Room 510, EC Building, 1001 Ta-Hsueh Road, Hsinchu City, Taiwan, ROC
Telephone: (03)5712121 x 54718
Fax: (03)5724176
Email: {fychong,fjwang}@csie.nctu.edu.tw

Abstract

Workflow technology has been frequently adopted by the applications of commerce, but only a few projects have been reported to consider workflow technology into non-business applications. In 2001, we constructed a workflow system to solve some traditional problems of departments in the universities. In order to deploy our workflow system to another department of similar request quickly, we propose an approach that constructs a *Process Model Template* and then reuses it. This paper describes what a process model template is and how to analyze and design a process model template. Finally, this paper demonstrates the implementation and reuse of a process model template.

Keywords: workflow, reuse, deployment, process model, template.

1. Introduction

In recent years, workflow technology has become more and more popular. Workflow management helps organizations to organize their work on different activities in such a way that predictable processes are carried out effectively and efficiently in a distributed environment [1]. The Workflow Management Coalition (WfMC) has developed a framework for the establishment of workflow standards [2]. In general, workflow technology is more frequently

adopted by the applications of commerce, since workflow technology can bring the benefits of business process modeling for specifying workflow-oriented application systems [3]. So far only a few projects have been reported to consider workflow technology into non-business applications [4].

With the achievement of a mature workflow management system developed in our laboratory, in 2001, we proposed a project that adopts our workflow technology to solve some traditional problems of departments in the universities. The project is now on the process of loading to work. An interesting phenomenon observed is that there are thousands of departments in the universities and the workflow system with each slight modification might be applied in a corresponding department. Without well strategies of reuse, system developers will waste a lot of time to construct a new system or modify their product to satisfy a department of similar request.

In this paper, we propose an approach that constructs a process model template used to construct a similar workflow application system quickly. This paper will describe the basic concept of a process model template and illustrate the construction process with four stages and four actions. This paper is organized as follows: Section 2 briefly introduces our workflow management system, and the workflow project. Section 3 describes what a process model template is, and illustrates how to analyze and design a process model template on our WFMS. Section 4 shows an example that demonstrates how to construct a real process model template and then reuses it. Chapter 5 is a conclusion of this paper.

2. Background

2.1 An overview of our WFMS

Our workflow management system, *Agentflow*, is completely based on n-tier software architecture. It includes four major components: 1) *PDE* (Process Definition Editor), a visual editor used to specify workflow processes, 2) *FormDesigner*, a graphical development tool

used to construct electronic forms, 3) *Agenda*, a client-side program of user interaction, and 4) *Flow Engine*, the core component of our WFMS.

When system developers want to build a workflow application on Agentflow platform, they use the PDE to write the specifications from all the process view, artifact view, and organizational view [5]. Moreover, two methodologies, process-oriented approach and artifact-oriented approach, are proposed for constructing workflow application systems [6]. Recently, workflow interoperability and system integration is becoming an important issue. Agentflow system also supports a fully interoperable and integrated environment. Our laboratory proposed an architecture using XML to integrate software applications [7]. In addition, our laboratory also proposed an approach to apply Enterprise Java Beans in an internet workflow management system [8]. These researches enrich the capability of Agentflow system.

2.2 The CSIE Workflow Project

The departments in the universities use traditional man-power resource to handle many works without office tools. They employ assistants or part-time students for these works. However, part of the works might be automated too. And flow-related works can be chosen to indicate the capability of Agentflow system. In 2001, we proposed a project to make a workflow service system for our department, namely CSIE workflow application system. This system involves five major sub-systems:

- Department-computer-center system
- Master-and-doctor-routine system
- Announce system
- Intra-laboratory system
- Agentflow-administration system

3. Constructing a Process Model Template on Agentflow System

3.1 Basic Concept and Our Approach

A project in Agentflow term is a process model designed for a specific application domain. A *Process Model Template* (PMT) is an incomplete process model made by removing specific environment parameters of an original process model. A PMT in our approach preserves the definition of control-flow and data-flow in an original process model. By inputting other personal environment parameters into a PMT, a complete process model for another similar domain can be obtained. For example, after constructing corresponding PMTs of the CSIE workflow application system, system developers can quickly construct another department workflow application system by reusing these templates.

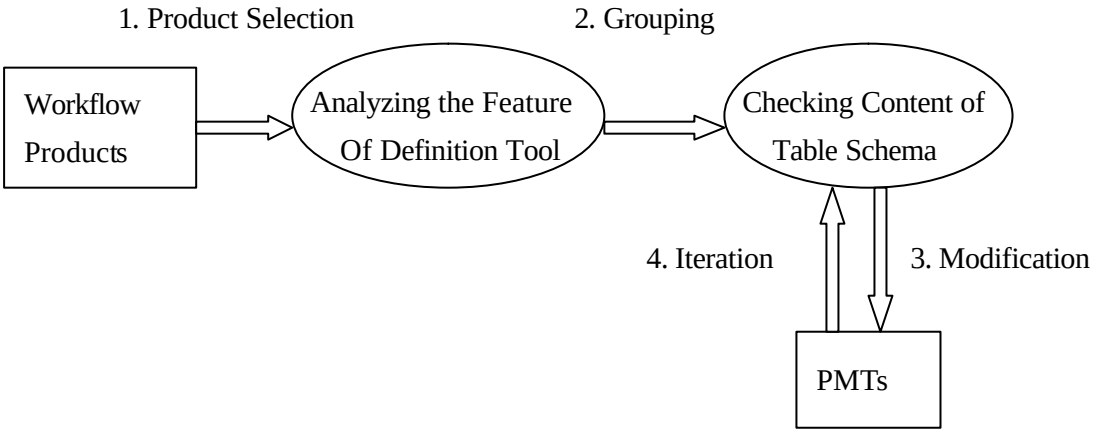

Figure 1. The PMT Construction Process

Figure 1 illustrates the PMT construction process. In our approach, the PMT construction process originates from the products and there are four actions that need be done:

1. **Product Selection:** Choose a product that has reusable value and economic benefit.
2. **Grouping:** Analyze the product and separate it into independent groups according to the feature of the process definition tool. Section 3.2 will describe the analysis of our process definition tool.
3. **Modification:** Observe the table schema of WFMS and modify the content of specific parameters in every independent group. Section 3.3 will describe the modification

methods.

4. Iteration: Return to 3rd stage again and re-modify the PMT, till no more change.

3.2 The Analysis of our Process Definition Tool

Our PMT is based on the process model of our WFMS. The analysis and design must suit the feature of our process definition tool, PDE, illustrated in Figure 2. The PDE includes four components: 1) *Project Information*, 2) *Activity Sub-Model*, 3) *Artifact Sub-Model*, and 4) *Project Role*.

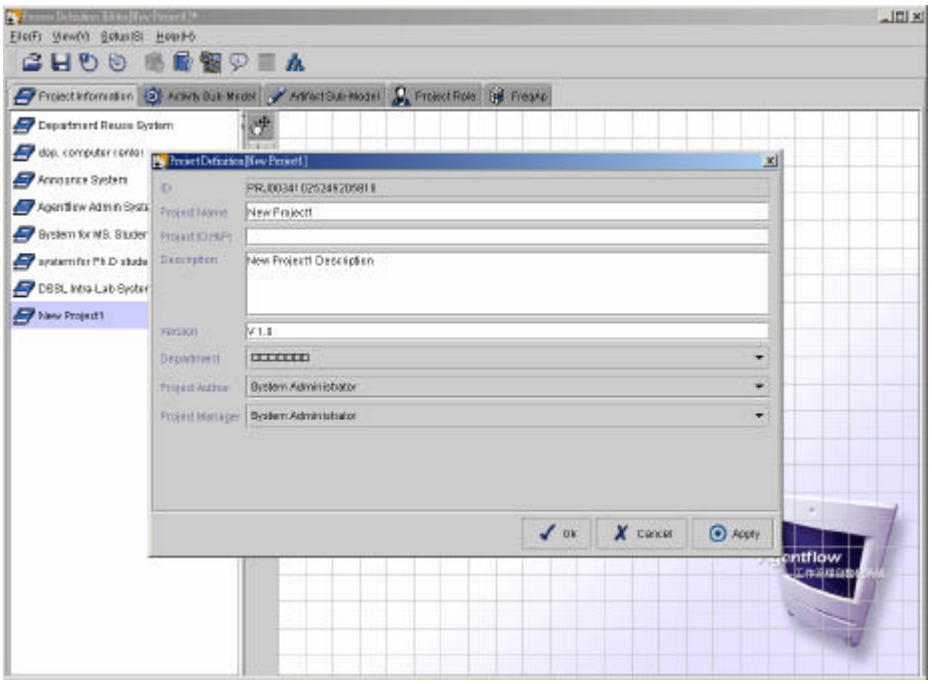

Figure 2. The PDE of Agentflow System

With implementation experience of the CSIE application system, there are some aspects needed to consider and check in the four components of PDE. Here are two different examples that modify an original workflow system to construct another similar workflow system. The first example is the Seminar-Paper-Insert process of the DSSL-Intra-Laboratory system. The following list is part of DSSL-Intra-Laboratory system specification:

- Project Name – DSSL Intra-Laboratory System
- Process Name – Seminar-Paper-Insert Process

- Execution Role – DSSL Master Students
- Artifact Name –Paper-Insert Form
- Runtime Script (Connection to Seminar Database) – IP= 140.113.x.x, Database= dssl, Account= sa, Password= 123

If one wants to reuse the DSSL-Intra-Laboratory system to other laboratory like PCSL, the following steps to modify the corresponding parameters which need be done:

1. Change the Project Name from DSSL to PCSL.
2. Reset the Master Students in their laboratory.
3. Reset the runtime script about seminar database information.

Another example is the Students-Personal-Study process of the Master-and-doctor-routine-work system. The following list is the part of its system specification:

- Project Name – Master-and-doctor-routine-work System
- Process Name – Students-Personal-Study Process
- Execution Role – 1. Office Assistant for this job, and 2. All Master Students
- The Title of Artifact – CSIE Personal-Study Form
- Time Control – The system will warn the students by sending an e-mail if this work does not complete after 7 days.

If one wants to reuse the Master-and-doctor-routine-work system to other department, the following steps to modify the corresponding parameters which need be done:

1. Reset the Office Assistant for this job in their department.
2. Change the title of CSIE Personal-Study Form.
3. Change the Time Control for their requirement.

According to above observation, there are five independent aspects needed to modify in general: 1) Process Information, 2) Execution Role, 3) Artifact Information, 4) Time/Event Control, and 5) Runtime Action. In order to design a PMT based on the process model of our WFMS, the table schema of Agentflow process repository must be understood and then the

specific parameters of an original process model must be checked and modified.

3.3 Five Independent Aspects of a PMT

3.3.1 Process Information

Process information supports basic data of a process. When users execute a process, they can read it. The related table schema about process information is `Project`, `Process_GenInf`, and `Process_Version`.

IDENTIFIER	MEANING
`Project_ID`	The key of this table. It is unique.
`Author_ID`	The `Member_ID` of author that has the privilege to modify the project. It is a foreign key from the `Member_GenInf` table.
`Project_Name`	The name of project used to be read by users.
`Project_Version`	The version of project used to distinguish this project from another similar project.

Table 1. The `Project` Table

Table 1 illustrates the `Project` table that records the project information. Since identifier `Project_ID` will be automatically created by Agentflow system, system developers can ignore it. Identifier `Author_ID` is the ID that has the privilege to modify the project. In a PMT, "System Administrator" should be selected to play as the role of author. Identifier `Project_Name` records project name, so system developers can save a general project name into a PMT, for example, "Announce System", but not save a specific project name, such as, "CSIE Announce System". Identifier `Project_Version` records project version. In a PMT, its default value is usually the same as the version of an original process model.

In order to make above description more systematically, these modifications can be distinguished as follows:

- No Modification (NM) – It means that system developers can ignore it and cannot change anything. For example, identifier "Project_ID" must be ignored.
- System Value Modification (SVM) – This modification means that system

developers must change original value to system value. For example, identifier "Author_ID" need be changed from original value to "System Administrator".

- General Value Modification (GVM) – This modification means that system developers must change specific value to general value. For example, the project name must be saved a general name.
- Flexible Value Modification (FVM) – This modification means that system developers can choose their personal value without restriction.
- Restricted Value Modification (RVM) – This modification means that system developers must save restricted value.

IDENTIFIER	MEANING
`Process_ID`	The key of this table. It is unique.
`Project_ID`	The `Project_ID` that the process belongs to. It is a foreign key from the `Project` table.
`Process_Name`	The name of process.
`Policy`	There are two kinds of dispatching policies: 1. random and 2. queue.
`AutoExec`	A Boolean string used to determine automatic execution or not.
`Aciton`	The Action-Script of process.
`PreAction`	The PreAction-Script of process.
`PostAction`	The PostAction-Script of process.

Table 2. The `Process_GenInf` Table

Table 2 illustrates the `Process_GenInf` table that records general information of process. Identifiers `Process_ID` and `Project_ID` belong to NM. Identifier `Process_Name` records process name and it belongs to GVM. Identifier `Policy` records dispatching policy and it belongs to FVM. Identifier `AutoExec` records a Boolean string used to determine automatic execution or not. It belongs to NM. Furthermore, identifiers `Action`, `PreAction`, and `PostAction` are relative to runtime action and they will be described in section 3.3.5.

IDENTIFIER	MEANING
Process_Version_ID	The key of this table. It is unique.
Process_ID	The Process_ID that the process version belongs to. It is a foreign key from the Process_GenInf table.
StartTime	The start time of process.
EndTime	The end time of process.
Executable	A Boolean string used to determine executable or not.

Table 3. The Process_Version Table

Table 3 illustrates the Process_Version table that records process version. Identifiers Process_Version_ID and Process_ID belong to NM. Identifiers StartTime and EndTime belong to FVM. The default value of identifier Executable is "false" and it belongs to RVM, since a PMT is impossibly executable.

3.3.2 Execution Role

Execution role defines the ones who can execute processes. In PDE, there are four methods provided for defining the process worker: Project role, Organization role, Role-from-process, and Role-from-artifact. The related table schema about execution role is Role_GenInf, Member_GenInf, Role_Member_Map, Process_Role_Map, Project_Role_GenInf and ProjectRol_Member_Map.

IDENTIFIER	MEANING
Role_ID	The key of this table. It is unique.
Role_Name	The name of role.
Department_ID	The department ID that the role belongs to. It is a foreign key.

Table 4. The Role_GenInf Table

IDENTIFIER	MEANING
Member_ID	The key of this table. It is unique.
Login_ID	The login ID of our WFMS.
Member_Name	The name of member.
Password	The login password of our WFMS.
Phone	The phone of member.
EMail	The e-mail of member.
Org_ID	The specific ID in their organization.
Main_Role_ID	The main Role_ID that member plays as. It is a foreign key from the Member_GenInf table.

Table 5. The Member_GenInf Table

IDENTIFIER	MEANING
Role_ID	The role ID. It is a foreign key from the Role_GenInf table.
Member_ID	The member ID. It is a foreign key from the Member_GenInf table.

Table 6. The Role_Member_Map Table

Table 4 illustrates the Role_GenInf table that records the general information of roles. Table 5 illustrates the Member_GenInf table that records general information of members. Table 6 illustrates the Role_Member_Map table used to map identifiers Role_ID and Member_ID. The Role_Member_Map table indicates the members who play the role in the Role_GenInf table. Agentflow system provides a graphical tool, Organizer, to construct the organization of company. System administrator can use Agentflow Organizer to edit these three tables. Usually, these tables are little modified for a long time, because they are relative to personnel transfer in an organization. It is not necessary to modify them in our approach.

IDENTIFIER	MEANING
Process_ID	The process ID. It is a foreign key from the Process_GenInf table.
Role_ID	The role ID. It is a foreign key from the Role_GenInf table.

Table 7. The Process_Role_Map Table

Table 7 illustrates the Process_Role_Map table used to map identifiers Process_ID

and Role_ID. It indicates the roles that have privilege to execute the process in the Process_GenInf table. In a PMT, it belongs to FVM, since there is no restriction about privilege setting.

IDENTIFIER	MEANING
Project_Role_ID	The key of this table. It is unique.
Project_ID	The Project_ID that the project role belongs to. It is a foreign key from the Project table.
Project_Role_Name	The name of project role.
ID	The specific ID in their organization.

Table 8. The Project_Role_GenInf Table

Table 8 illustrates the Project_Role_GenInf table that records project role general information. Identifiers Project_Role_ID and Project_ID belong to NM. Identifier Project_Role_Name records the name of a project role and it belongs to GVM. Identifier ID belongs to FVM.

IDENTIFIER	MEANING
Project_Role_ID	The project role ID. It is a foreign key from the Project_Role_GenInf table.
Member_ID	The member ID. It is a foreign key from the Member_GenInf table.

Table 9. The ProjectRole_Member_Map Table

Table 9 illustrates the ProjectRole_Member_Map table used to map identifiers Project_Role_ID and MemID. It indicates the members who play the project role in the Project_Role_GenInf table. In a PMT, these records need to be cleared, since different organizations have different setting of project roles. It belongs to RVM.

3.3.3 Artifact Information

Many tasks need to operate some data such as documents, reports, records, etc. In AgentFlow System, artifacts are used to capture and collect these data. Artifact information represents formal information of an artifact, including artifact name, synopsis, serial number…etc. The related table schema about artifact information is Artifact_GenInf,

and “ARTIFACT”+Artifact_ID+“_Form”.

IDENTIFIER	MEANING
Artifact_ID	The key of this table. It is unique.
Project_ID	The Project_ID that the artifact belongs to. It is a foreign key from the Project table.
Artifact_Name	The name of artifact.

Table 10. The Artifact_GenInf Table

Table 10 illustrates the Artifact_GenInf table that records general information of artifacts. Identifiers Artifact_ID and Project_ID belong to NM. Identifier Artifact_Name records artifact name and it belongs to GVM.

IDENTIFIER	MEANING
Item_ID	The item ID inside this artifact. It is unique.
Item_Name	The name of item.
DefValue	The default value of item.
Property	The property of item.

Table 11. The “ARTIFACT”+Artifact_ID+“_Form” Table

Table 11 illustrates the “ARTIFACT”+Artifact_ID+“_Form” table used to record attributes and plug-in components of an artifact. In a PMT, identifiers Item_ID and Item_Name are the same as its original process model, so they belong to NM. There is no restriction about identifier DefValue, so it belongs to FVM. Identifier Property records attributes of an item, such as, size, location, color, text, script and so on. In a PMT, it belongs to GVM.

3.3.4 Time/Event Control

In real world, many tasks run on a regular time schedule, or start after some events have happened. So, the time/event control is needed in a workflow management system. Agentflow system can let developers set system events to the deadline of each process. The related table schema about time/event control is Pro_Warning and CronTable.

IDENTIFIER	MEANING
Process_Warning_ID	The key of this table. It is unique.
Process_ID	The Process_ID that the warning belongs to. It is a foreign key from the Process_GenInf table.
After_Time	The leaving minutes to execute the warning action after process starting.
Function	There are six warning functions: 1. wake up myself, 2. wake up my manager, 3. wake up myself by e-mail, 4. wake up my manager by e-mail, 5. bypass, and 6. user's script definition.

Table 12. The Process_Warning Table

Table 12 illustrates the Process_Warning table used to handle the time/event control. Identifiers Process_Warning_ID and Process_ID belong to NM. In a PMT, there is no restriction about process warning. System developers can flexibly save null or some default values into identifiers After_Time and Function, so they belong to FVM.

IDENTIFIER	MEANING
Process_ID	The Process_ID of process that will be executed automatically and periodically by the system. It is a foreign key from the Process_GenInf table.
Entry	The entry used to check the execution time.

Table 13. The CronTable Table

Table 13 illustrates the CronTable table used to indicate the process that will be executed automatically by the system at the time entry. In a PMT, system developers can flexibly save null or some default values into identifiers Process_ID and Entry, so they belong to FVM.

3.3.5 Runtime Action

Agentflow system supports intermediate language to describe complex program logic for runtime action. The context of this script language is similar to Java script, and system developers can use some system objects and functions to retrieve process definition and control the flow. The related table schema about runtime action is Process_GenInf, Process_Warning, and "ARTIFACT"+Artifact_ID+"_Form". System developers

can use Agentflow Script Editor to write script program.

In a PMT, system developers should remain the script's computing logic of an original process model, but not the specific parameters or contexts. The following example shows a script that connects to a database and retrieves some records of Name column from the Seminar table.

```
Packages.java.lang.Class.forName("com.inet.tds.TdsDriver");
var con = Packages.java.sql.DriverManager.getConnection
("jdbc:inetdae:140.113.210.11:1433?database=dssl&sql7=true&charset=BIG5","sa","123");
var myStatement = con.createStatement();
var sql = "select distinct Name from Seminar";
var rs = myStatement.executeQuery(sql);
var cb_name = Form.getComponent("CB_name");
while(rs.next())
{
        var name = rs.getString("name");
        cb_name.addItem(name);
}
```

In this example, the specific parameters such as IP address, port, account, and password need be removed. Here are two methods to modify these parameters or contexts:

1. *String Replacing* – System developers could change specific parameters or contexts to special string patterns, for example, they can change IP address from "140.113.210.11" to "db_ip". When reusing the template of this kind, they need to replace these string patterns to the parameters or the contexts of their personal specification. This method will be easy but not flexible.
2. *Dynamic Loading* – System developers could create additional tables to save specific parameters or contexts, and write additional scripts used to retrieve these parameters or contexts from those tables. When reusing this template of this kind, they just need to insert their personal parameters or contexts into the corresponding tables. This method will be flexible but not easy.

4. An Example

4.1 Constructing the PMT of Intra-Laboratory System

In this section, the DSSL-Intra-Laboratory system is considered as the original process model used to construct our PMT of Intra-Laboratory system. The following parts show the modifications of this template:

- **Process Information Modification**

Project_ID	Author_ID	Project_Name	Project_Version
001	fychong	DSSL-Intra-Laboratory System	V1.0

Table 14. The Project Information of the DSSL-Intra-Laboratory System

Process_ID	Project_ID	Process_Name	Policy	AutoExec
P0001	001	Seminar-Paper-Insert process	Random	false
P0002	001	Seminar-Paper-Delete process	Random	false
P0003	001	Seminar-Paper-Query process	Random	false

Table 15. The Process General Information of the DSSL-Intra-Laboratory System

Process_ID	StartTime	EndTime	Executable
P0001	2002-02-22	*	true
P0002	2002-02-22	*	true
P0003	2002-02-22	*	true

Table 16. The Process Version of the DSSL-Intra-Laboratory System

Table 14, Table 15, and Table 16 show the process information of the DSSL-Intra-Laboratory system and the following steps need be done:

Step 1) Change the project name from "DSSL-Intra-Laboratory System" to "Intra-Laboratory System".

Step 2) Change the author from "fychong" to "System Administrator".

Step 3) Change the Executable of process version from "true" to "false".

- **Execution Role Modification**

Role_ID	Project_ID	Role_Name	Org_ID

PR001	001	DSSL Instructor	Null
PR002	001	DSSL Doctor Student	Null
PR003	001	DSSL Master Student	Null

Table 17. The Project Role General Information of the DSSL-Intra-Laboratory System

Role_ID	Member_ID
PR001	M0012345
PR002	M8717544
PR002	M8817519
PR003	M8917512
PR003	M8917566
PR003	M9017581

Table 18. The Content of the `ProjectRole_Member_Map` Table of the DSSL-Intra-Laboratory System

Table 17 and Table 18 show the project role information of the DSSL-Intra-Laboratory system and the following two steps need be done:

Step 1) Change the project role name from “DSSL Instructor”, “DSSL Doctor Student”, and “DSSL Master Student” to “Instructor”, “Doctor Student” and “Master Student”.

Step 2) Remove all records that RolID equals “PR001”, “PR002”, or “PR003” from the PrjRol_Mem table.

- **Runtime Action Modification**

In the DSSL-Intra-Laboratory system, all of the three processes have the following script that connects to a database and then inserts, deletes, or queries the data of seminar papers.

```
var con = Packages.java.sql.DriverManager.getConnection
("jdbc:inetdae:140.113.210.11:1433?database=dssl&sql7=true&charset=BIG5","sa","123");
```

Here are the two methods in our approach used to modify these parameters:

1. String Replacing Method:

Step 1) Change “140.113.210.11” to “db_ip”.

Step 2) Change “1433” to “db_port”.

Step 3) Change “dssl” to “db_name”.

Step 4) Change “sa” to “db_account”.

Step 5) Change “123” to “db_password”.

After modification, the script will be like as:

```
var con = Packages.java.sql.DriverManager.getConnection
("jdbc:inetdae:db_ip:db_port?database=db_name&sql7=true&charset=BIG5","db_account","db_p
assword");
```

2. Dynamic Loading Method:

At first, an additional table, called `DB_Parameter` illustrated in Table 19, need be created and be saved these parameters.

IDENTIFIER	MEANING
ID	The key of DB_Paramter table. It is unique.
DB_IP	The IP address of database server.
DB_Port	The port of database server.
DB_Name	The database name.
DB_Account	The system account of database.
DB_Password	The password of system account.

Table 19. The `DB_Parameter` Table

Moreover, system developers need to modify the original script to the following example that retrieves parameters from the DB_Parameter table.

```
var SQLText = “Select * from DB_Parameter where ID = 1”;
var DataSet = Client.SQLloadValue(SQLText);
var sRecord = DataSet.get(0);

var IP = sRecord.get(”DB_IP”);
var PORT = sRecord.get(”DB_Port”);
var DBNAME = sRecord.get(”DB_Name”);
var ACCOUNT = sRecord.get(“DB_Account”);
```

```
var PASSWD = sRecord.get(“DB_Password”);

var con = Packages.java.sql.DriverManager.getConnection
("jdbc:inetdae:IP:PORT?database=DBNAME&sql7=true&charset=BIG5","ACCOUNT","PASSWD");
```

There is no modification in the artifact information and time/event control. After operating the modifications mentioned above, a PMT of Intra-Laboratory system can be produced. Furthermore, system developers can use the PDE to save and export this template to a “.PRJ” file. When they want to reuse this template in the future, they just need to import this “.PRJ” file in the beginning.

After constructing a PMT, system developers can use any programming tools that have the capability to insert their personal parameters into the table schema, for example, pure java, ASP, JSP, Visual C++, and so on. There is no restriction about deployment tools.

4.2 Deploying the Template

In this section, we will use Agentflow system to create a deployment process used to deploy the PMT of Intra-Laboratory system. This deployment process contains two sub-processes: 1) Laboratory-Information-Setting process, and 2) Seminar-Database-Setting process.

First, the Laboratory-Information-Setting process is used to set the laboratory name, instructor, doctor student, and master student. Figure 3 illustrates the inputting diagram in Agenda. After submitting this form, the Laboratory-Information-Setting process will get the value from this artifact, and then update the process information and execution role aspects in the PMT of Intra-Laboratory system. This sub-process will update the following three tables: `Project`, `Project_Role_GenInf`, and `ProjectRole_Member_Map`.

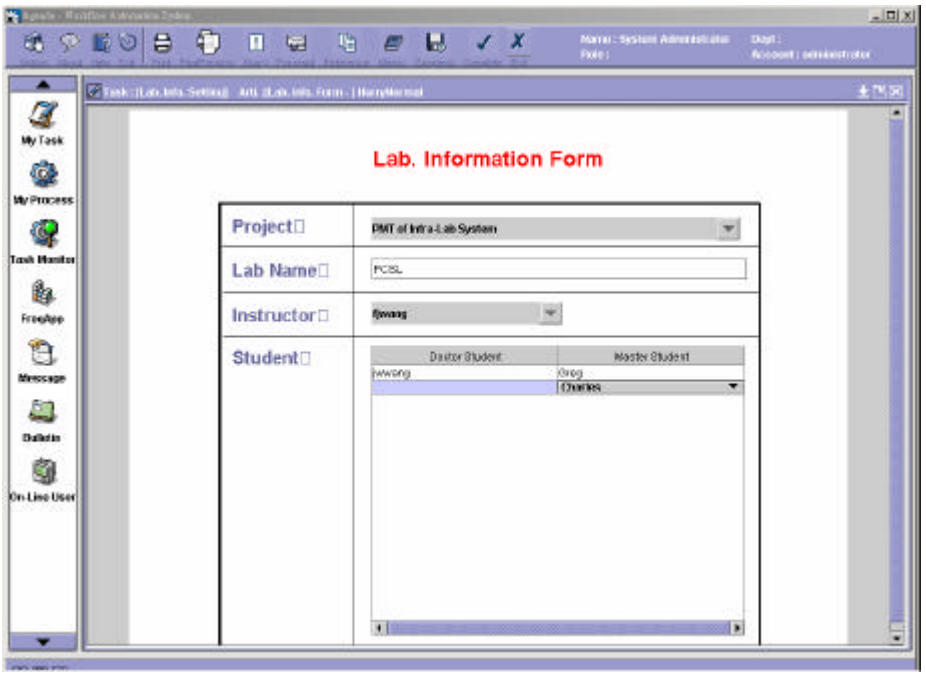

Figure 3. The Inputting Diagram of Laboratory-Information-Setting Process

Second, the Seminar-Database-Setting process is used to set the database information of seminar. Figure 4 illustrates the inputting diagram in Agenda. After submitting this form, the Seminar-Database-Setting process will get the value from this artifact and create an additional table, called Seminar, used to save the seminar data. Then, the process will use the value to update the runtime action aspect in the PMT of Intra-Laboratory system. If the runtime action of this PMT is modified by the string replacing method, users need to replace them again. If the runtime action of this PMT is modified by the dynamic loading method, users just need to insert the value into the `DB_Parameter` table.

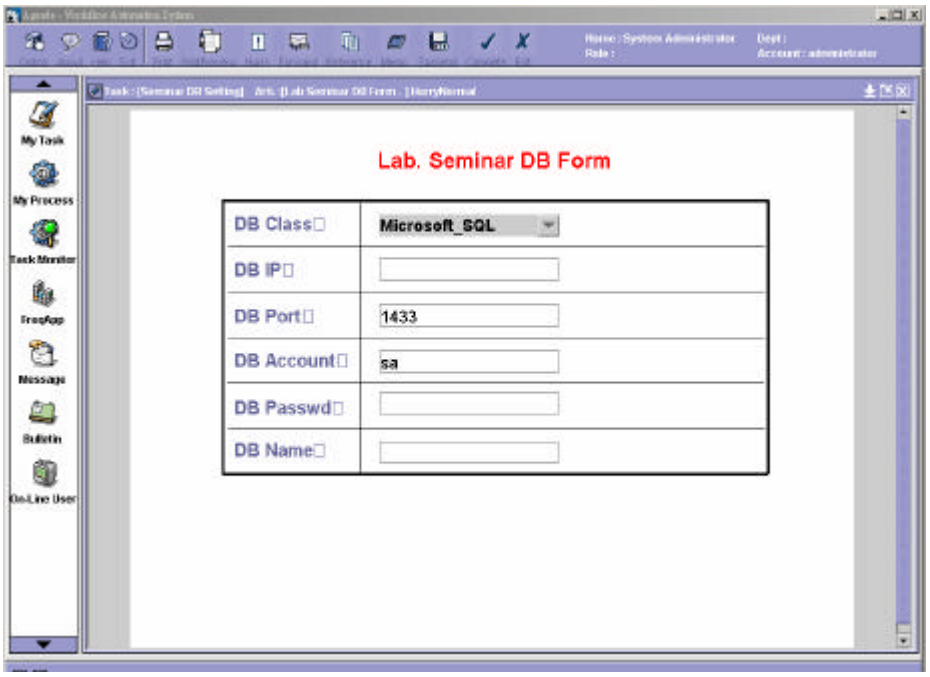

Figure 4. The Inputting Diagram of Seminar-Database-Setting Process

5. Conclusion

We have presented an approach that constructs a PMT based on the process model of our WFMS and reuse it. A PMT in our approach is divided into five independent aspects and there are five modification actions to modify specific parameters in our table schema. We also have demonstrated an example that implements a PMT of Intra-Laboratory system and then reuses it. The examples indicate that our approach allows the following benefits:

1. Without the help of workflow management system.
2. More efficient than specifying with workflow management system directly.
3. With a friendly interface for building whole system.
4. Easily setup all parameters step by step.

There are two future works needed to be paid attention. The first one is the issue of flexibility. A PMT in our approach is not flexible, because of the preservation of definition of control-flow and data-flow. The second one is the issue of constructing a standard library of PMT. In order to manage and share our PMTs effectively and efficiently, a standard and big library is expected.

Reference

[1] Cugola, G.; Di Nitto, E.; Fuggetta, A. "The JEDI Event-Based Infrastructure and Its Application to the Development of the OPSS WFMS". *Software Engineering, IEEE Transactions*, Volume: 27 Issue: 9 , Sept. 2001.

[2] WFMC Workflow Standard – *Workflow Reference Model.* http://www.wfmc.org/standards/docs/tc003v11.pdf

[3] Michael Amberg. "The Benefits of Business Process Modeling for Specifying Workflow-Oriented Application Systems". http://www.seda.sowi.uni-bamberg.de/forschung/publikationen/html/wfmc-cfp.htm

[4] Reuss, T.; Vossen, G.; Weske, M. "Modeling Samples Processing in Laboratory Environments as Scientific Workflows". *Proceedings of Database and Expert Systems Applications*, Eighth International Workshop, 1997.

[5] Y. S. Chen. *Building an Edit and Enactment System for Process Definition*. Master Thesis, National Chiao-Tung University, 2001.

[6] R. J. Liou. *Constructing Workflow Application Systems*. Master Thesis, National Chiao-Tung University, 2001.

[7] S. Z. Yeh. *Integrate Workflow System on Internet with XML*. Master Thesis, National Chiao-Tung University, 2001.

[8] D. J. Lin. *Applying Enterprise JavaBeans in an Internet Workflow ManagementSystem.* Master Thesis, National Chiao-Tung University, 2001.