Using Software Agents to Design a Modern WfMS

Jian-Wei Wang, Chiung-Wen Chang and Feng-Jian Wang
Department of Computer Science and Information Engineering
National Chiao Tung University, Hsinchu, Taiwan, ROC
{jwwang,changcw,fjwang}@csie.nctu.edu.tw

Abstract*-Traditional workflow management systems (WfMS) are based on a centralized architecture, which need be modified to suit for the Internet environment. Their workflows are restricted to pre-defined resource allocation constraints, and the routing decisions of flow processes cannot be changed easily and dynamically. Besides, it is also not flexible for the traditional systems to provide suitable views and operations for different participants or roles. This paper introduces agent technologies into WfMSs and presents a new WfMS architecture that could provide adequate and flexible solutions to cope with these shortcomings.*

Keywords: Workflow, Workflow Management System, Internet, Agent

1. Introduction

Workflow management came mainly from the domain of office automation and this technology has evolved for decades. With the evolution of Internet technology, the requirements of workflow also increase in various applications. Traditional WfMS technology needs large more efforts to support these demands. The defects can be discussed with four significant aspects[1][2].

- Scalability-The traditional WfMS is basically a centralized architecture, which may not be scalable for the current trends.
- Extensibility-The business process scenarios and data representations are greatly restricted at design time. The adopted model cannot be easily extended to meet new requirements and standards.
- Flexibility-Traditionally, initiation and completion conditions of each process activity can be statically described only. Artifact representations and routings are also defined in strict and limited rules.
- Adaptability-The enactment services of current WfMS are usually restricted by predefined flow definitions. When suffering unexpected cases, the execution of workflows lacks dynamicity in task scheduling and dispatching.

From the agent's viewpoint, the business process scenarios are better modeled as the interactions among the system components, users, and software agents. Namely, the traditional software modules can be assisted or even accomplished by software agents completely. Here we adopt the major characteristics of software agents such as mobility, autonomy, and intelligence to construct, simplify and empower the whole workflow system and its applications.

The agents adopted by our system are divided into three major types. *Process Agents* facilitate managing and monitoring the enactment of workflow instances. *Artifact Agents* are designed to take charge of the artifact data transmission and tasks accomplishment. *Person Agents* assist participants in accomplishing tasks and are also responsible for artifact presentation on the Client Tool. Furthermore, our agents can perceive the changes of external environments, so that they may adjust themselves timely and adequately based on predefined rules.

In our system architecture, a process agent acts as an independent service executor in the distributed environment. Each workflow instance is carried out by an individual process agent exactly. Next, the database access of artifact agents can be deemed different from that of the centralized way. Each artifact agent can carry required data and migrate to designated places for computation. Subsequently, person agents act as participant users themselves. Their autonomy characteristics bring great impacts on client tools and user operation habits. That is, the client tool can behave more actively to give notices or report events to users.

The rest of this paper would be organized as follows. Section 2 introduces recent research works on agent-oriented workflow systems. Section 3 clearly describes our system architecture. In section 4, a workflow example is given to explain the actual works of the system. Section 5 concludes with the advantages of the proposed system architecture and future works.

2. Related Research Works

Owing to the increasing shortcomings of the traditional WfMS, many researchers in the related area have invested a lot of time in enhancing the workflow management and proposed various modified WfMS architecture. Researches about

introducing the software agent technology into the WfMS have been also invested for years. On typical related project is ADEPT[3] project proposed by British Telecom Lab., which focuses on enhancing the supply-chain management. The system consists of multiple software agents the concurrently negotiate an agreement on how resources should be assigned to support a business process. For another example project, Leangzhao Zeng[4] et al. proposed an approach that combines agents and workflows to effectively integrate cross-enterprise workflows. It is in order to support virtual enterprise and business-to-business e-commerce.

These researches are all emphasized on inter-organization workflows, such as supply-chain management and business-to-business processes. Although the service-oriented architecture (SOA) has been proposed in order to solving B2B and B2C workflow behaviors, these workflows are relatively simple and fixed. Few researches take a complex user/role model into consideration. So in this paper, we aim at presenting an agent-based workflow system for primitive workflows with an extended process modeling.

3. System Architecture

3.1. System Architecture – Design Phase

The activities in the design phase of our entire system are shown in Figure 1. The top half of the figure is primarily refereed to the PLAN(Process LANguage Model[5][6] and the system design of Agentflow[7]. In this paper, we extend the PLAN model and divide it into three sub-models named process, role, and artifact individually.

The *process sub-model* describes the functional and behavioral perspectives of a software process. It includes the methods of workflow initiation, completion and termination. It also models various workflow behaviors during workflow enactment. Besides, we add the time constraints for design into the process sub-model: workflow designers can destine the time about the accomplishment or initiation of workflows and replacement of newer version workflows. The *role sub-model* illustrates the organizational perspective of a software process. It consists of the organization hierarchy, personnel property and communication among the roles. The *artifact sub-model* illustrates the informational perspective of a software process. It demonstrates the reference materials such as forms, documents and programs and the lifecycle (described in states) of an artifact. An artifact is defined as a product referred or generated by a workflow.

There are also three corresponding tools in our system, the *Process Designer Environment*, *Organization Designer Environment* and *Artifact Designer Environment*. By using the graphics modules and visualization tools provided by these designer environments, workflow designers could specify and design a workflow more flexibly and efficiently.

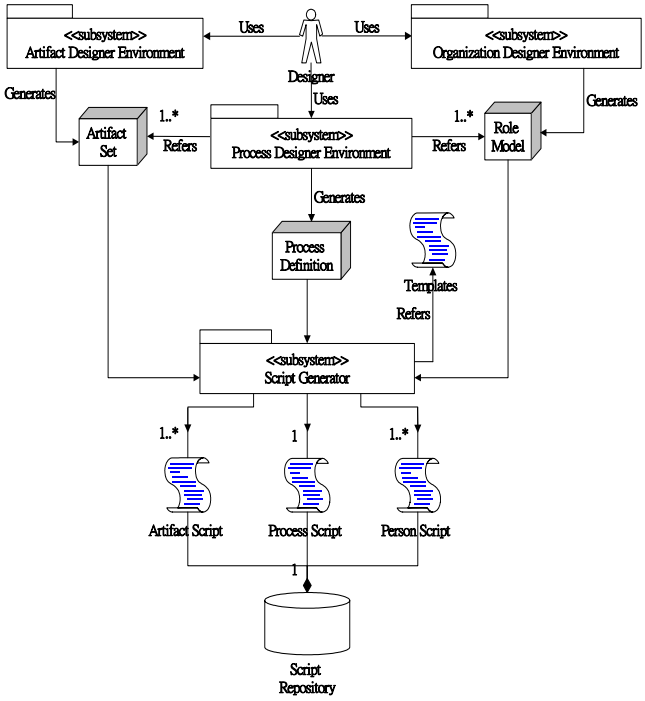

Figure 1 System Architecture – Design Phase

When the definitions of the entire workflow are completed, supporting environment would gather the documents and organizational information and stockpile them into the Role Model and Artifact Set individually. In addition, it also groups the information about the workflow enactment, such as rules, events, and constraints into the Process Definition. These three models are all described with the XML format for easy processing. With XML, our system is more convenient and feasible to integrate external applications and agent systems, or to interchange process definitions with other process model languages such as XPDL[8] advised by WfMC[2].

Subsequently, the *Script Generator* is responsible for analyzing these workflow definitions and then generating the agent script files, which are stockpiled into the *Script Repository*. The generation is done by referring to the pre-designed templates and using respective components. And the Script Generator and Templates are constructed according to the agent platform adopted. These modules designed increase the extensibility of our system. Therefore, an application workflow designed inside this phase is translated into executable agent script files.

3.2. System Architecture – Execution Phase

The execution phase inside our system is shown in Figure 2. There are some important components, such as the *Script Repository* for storing scripts and

the *Agent Manager* for instantiating, destroying and monitoring agents shown in the figure. The primary reason for separating the agent manager from the workflow manager is for flexibility. That is, the agent manager is constructed dependent on the agent platform adopted. It provides not only the services, such as naming, instantiation, destruction and location etc., but also a common set of API calls for the workflow manager.

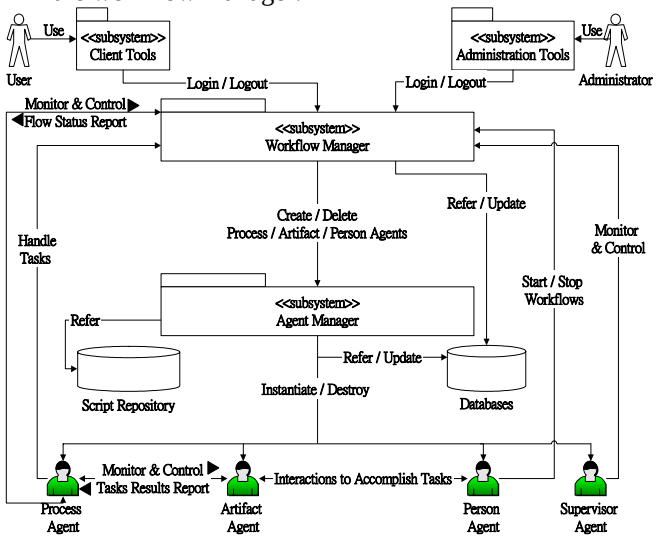

Figure 2 System Architecture – Execution Phase

Operators, which include users and administrators, use the *Client Tool* or *Administration Tool* to login our system. Besides, a supervisor can monitor and manage all running workflows by the Administration Tools, but an employee can only watch the workflows he instantiates or handles. In the next subsections, we will describe the person agent, artifact agent, process agent and workflow manager thoroughly with their designs and cooperation.

3.2.1. Person Agent

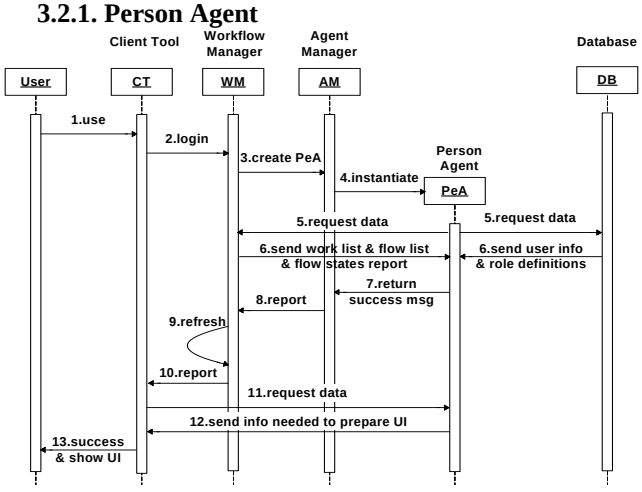

Figure 3 Instantiation of a Person Agent

A person agent is an intelligent agent that acts as an interface between the system and the user. It represents the user interface function in a workflow system and facilitates related workflow operations. A person agent is instantiated by the agent manager when the user is online, and destroyed when the user leaves. The Supervisor Agent in Figure 2 is actually a kind of person agent; it is instantiated when the administrator logins our system by the Administration Tool. Figure 3 below depicts the instantiation of a person agent.

The profile of a person agent is based on the Role Model. The information retrieved for a person agent includes contents and capabilities when the person agent is instantiated. The information is described below.

(1) Contents

- Intenal-record: It contains an agent-id and the information about the corresponding user. The information includes name, phone, email and customized configuration, etc.
- Role: The role represents a position where a user occupies in the organization.

(2) Capabilities

- Role-specific user interface: A person agent provides a role-specific interface according to the customized configurations of a corresponding role.
- Workflow management: A work list is a set of tasks derived from the workflow manager and needs to be accomplished by a user. The tasks may be properly scheduled. With this viewpoint, a person agent can act as a work list handler. When a user takes a task, the corresponding person agent can provide a proper user interface such as e-forms[7], which are carried by artifact agents.
- Flow list management: A flow list is a set of workflows that can be worked by the user. It is also dependent on the user's role(s) and related workflow enactment constraints. When a user wants to enact a workflow, he picks it from the flow list in the user interface and sends an instantiation request to the workflow manager through his corresponding person agent.
- Flow state report management: The report records the states of each workflow which has been worked by the user. The report includes the flow state such as finish, waiting, running, and suspended. With the aids of reports, a user can monitor the workflow progress he worked.
- Communication: The communication targets are the process agent, artifact agent and workflow manager. Firstly, a person agent needs to interact with a process agent to retrieve the necessary data for detailed workflow progress representation. Secondly, a person agent interacts with an artifact agent for

user interface representations. Lastly, a person agent retrieves the relevant data such as work list and flow list from the workflow manager.

3.2.2. Artifact Agent

An artifact agent is a mobile agent that acts as a data carrier, which migrates between sites and interacts with its process agent and person agents to accomplish the task. An artifact agent is instantiated by the agent manager when the process agent prepares to execute the tasks of a workflow, and destroyed when all tasks are accomplished. There may be more than one artifact agent, and the number of artifact agents of a workflow depends on the route.

Figure 4 below depicts the lifecycle of an artifact agent clearly. And an artifact agent can repeat the actions numbered 8 to 10 before being destroyed. Furthermore, before receiving the request of a person agent, an artifact agent usually resides at the site it is generated for being together with the process agent, or locates itself in the task list owned by the workflow manager

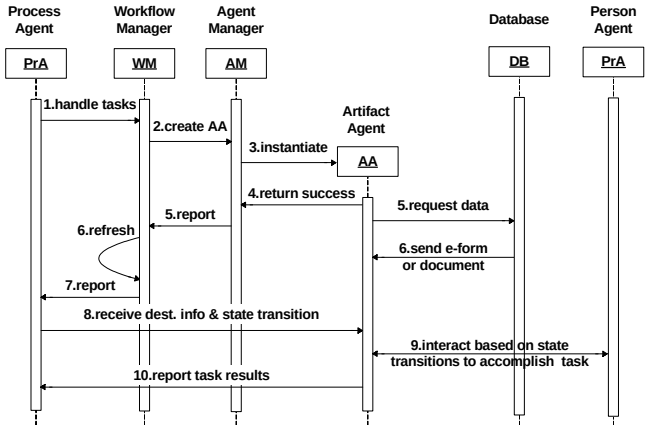

Figure 4 Lifecycle of an Artifact Agent

The profile of an artifact agent is retrieved from the Artifact Set and/or databases. The information contained in the profile can be divided into two parts.

(1) Contents

- Internal-record: It contains an agent-id and the carried artifacts such as e-forms, documents or other data, which are necessary to accomplish the task.
- State-transition: It contains the rules to interact with person agents to accomplish the task. The rules can be obtained from its process agent. Note that the information about the lifecycle of an artifact agent is kept in its process agent. The state transition here is also helpful to the artifact representation.

(2) Capabilities

- Communication: The communication targets are its process agent, person agents and the workflow manager. Firstly, an artifact agent needs to report task results and get the next route data from its process agent. Secondly, an artifact agent provides necessary artifacts to a person agent for user interface representations and receives the user's decisions from his person agent. Lastly, an artifact agent asks the workflow manage to report an online agent-id satisfied the conditions for executing the task.
- Mobility: An artifact agent could migrate between the sites indicated by its process agent.
- Offline-handling: If the target person agent does not exist (i.e. user is offline), an artifact agent returns and reports to its process agent. If there is another choice, the artifact agent would migrate to the new destination accordingly. Otherwise, the artifact agent will queue itself into the task list of the workflow manager.

3.2.3. Process Agent

A process agent is an intelligent agent that acts as a workflow enactment service executor in the traditional WfMS. For example, a process agent is instantiated when a workflow is enacted by the user or the system, and destroyed when the workflow is accomplished or canceled.

The profile of a process agent is based on the Process Definition. The information contained in the profile can be divided into two parts.

(1) Contents

- Internal-record: It contains an agent-id and the workflow data including name, description, starter, state time and due time etc.
- Rules: The rules contain the information needed to enact the workflow respectively.

(2) Capabilities

- Artifact agent management: A process agent can instantiate and maintain enough artifact agents to work. In the meanwhile, the process agent will monitor and interact with these artifact agents. When the workflow associated with an artifact agent is accomplished or canceled, the process agent then destroys the agent.
- Routing decision: A process agent can determine the routing paths based on the internal predefined rules, previous task results and by perceiving related external environment changes.
- Communication: The communication targets are the workflow manager, its artifact agent and person agent. Firstly, a process agent reports the workflow states to the workflow manager after its artifact agent(s) migrates to next node. Secondly, a process agent gives the next route data including roles and constraints etc., and state transition to its artifact agent according to current workflow state, and receives enacted task results from its artifact agent. Lastly, a process agent needs to interact with a person

agent for preparing detailed workflow execution progress shown on the client tool.

3.2.4 Workflow Manager

The workflow manager, as implied by the name, is a component that manages all executable workflows in our system. It also maintains records of all running workflow instances in the system and updates them persistently. Besides, both the instantiation and destruction of process agents, artifact agents or person agents rely on the instructions of the workflow manager. Figure 5 below depicts the interactions of the workflow manager with other system components clearly.

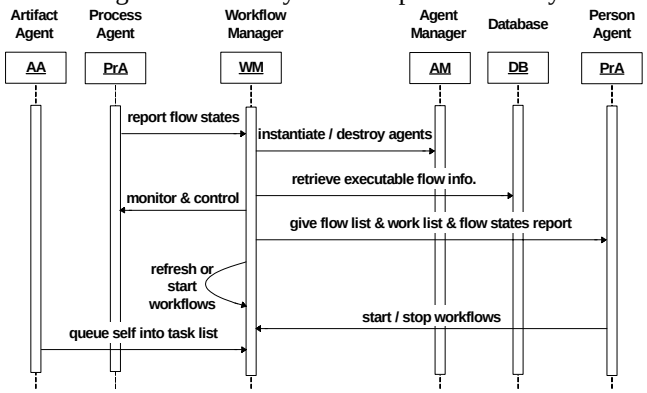

Figure 5 Interactions of the Workflow Manager

The profile of the workflow manager is loaded from the database and would be persistently updated. It can be divided into two parts.

(1) Contents

- Executable flow: It contains all available workflow definition for enactment in the system currently.
- Flow State: It contains the states of all enacted workflow instances (i.e. process agents) in the system currently.
- Task list: It contains the work list of all users in the system, i.e., the relationship of artifact agents and person agents.
- Agent list: It maintains a record of agent-id of all running artifact agents and person agents.

(2) Capabilities

- Process agent management: All the instantiation and destruction of workflows are monitored and maintained by the workflow manager. After completing the work in one site, the process agent reports its workflow state to the workflow manager. Besides, when a user's request for enacting or stopping a workflow is accepted, the workflow manager is responsible for instantiating or destroying the corresponding process agent.
- Task list management: There exist two situations for updating the task list. When the designated person agent of an artifact agent is offline, the artifact agent will queue itself into the task list. When the user appears to logout the system, those artifact agents, whose tasks have not completed yet, will queue themselves into the task list.
- Agent list management: Both the instantiation and destruction of artifact agents and person agents are settled by the workflow manager. That is, requests from the process agent or client tool will be propagated to the agent manager by the workflow manager. So the workflow manager can keep and maintain the lists of artifact agents and person agents.
- Communication: The communication targets can be the process agent, person agent, agent manager and artifact agent. Firstly, the workflow manager gets flow states from reports of process agents and persistently updates them. Secondly, the workflow manager gives flow list, work list and flow states report to a person agent. Thirdly, the workflow manager would ask the agent manager to instantiate or destroy agents. Lastly, the workflow manager reports the satisfied agent-id to an artifact agent for executing the task.

4. An Example

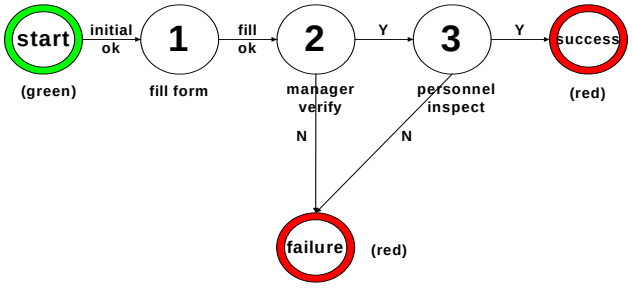

Figure 6 A Simple Ask-for-leave Workflow

Here we use a simple ask-for-leave workflow as the scenario example to describe the handing of a workflow instance being enacted.

The ask-for-leave workflow has three process nodes as shown in Figure 6. And we assume there is an employee called Aaron who wants to ask for leave. Firstly, Aaron must login our system to use the client tool, and his corresponding person agent, PeA_Aaron, is instantiated in the agent platform after password authentication. Secondly, he enacts the ask-for-leave workflow from the flow list provided by the client tools. Once PeA_Aaron receives the enactment instruction, it asks the workflow manager to create the corresponding workflow instance. Subsequently, the process agent, PrA_leave, will be instantiated to take charge of the enactment of the ask-for-leave workflow.

When PrA_leave is successfully instantiated, it firstly asks the workflow manager to create an artifact agent to carry the required workflow data. When the artifact agent, AA_leave, is instantiated successfully, it then retrieves the data objects, which compose an e-form, from the database. According to the workflow definition, Aaron's person-id is sent to AA_leave by PrA_leave. Next, AA_leave asks the workflow manager to report the target agent-id, so PeA_Aaron will be notified about the new item added to the work list and ask the client tool to show the change.

When Aaron picks up the ask-for-leave to execute, PeA_Aaron notifies AA_leave to migrate to this site and interacts to decide how the data objects should be shown. After Aaron completes the filling, AA_leave migrates back to report the new state to PrA_leave. Next, PrA_leave analyzes the state and determines the next route. According to the workflow definition, the next stage is the manager verification and its target operator is the manager of the form writer. Assume that there are two candidates called Bob and Cindy, but Cindy is offline currently. Then AA_leave will receive the agent-id of Bob's person agent.

Again, there will be a new item called ask-for-leave added on Bob's work list. When Bob executes this task, AA_leave receives the notice of Bob's person agent and migrates to this site to interact for adequate artifact representation. Now the first and second parts of the e-form need to be shown, but the first part is now read-only. When Bob completes the task, AA_leave migrates back to report the result to PrA_leave. If the decision is "rejection", PrA_leave detects the flow is going to complete. It sends the rejection result message to the workflow manager to notify the flow is going to complete. Subsequently, the workflow manager notifies PeA_Aaron the rejection result and asks the agent manager to destroy AA_leave and PrA_leave.

If Bob's decision is "acceptance", PrA_leave confirms that the next route is the personnel inspection. The target operator is restricted to the members of personnel department only. Except the artifact representation, the following sequences are similar to the previous stage mentioned above. Now the e-form shown to the operator includes all three parts and the first two parts cannot be modified.

Assume the task of personnel inspection is completed, and the decision is "acceptance". PrA_leave receives the result and detects the flow is going to complete. Then, it sends the acceptance result to the workflow manager and notifies the latter that flow is going to complete. Subsequently, the workflow manager notifies PeA_Aaron about the acceptance result and then asks the agent manager to destroy AA_leave and PrA_leave. Finally, Aaron gets the result of the ask-for-leave workflow and the whole execution of an ask-for-leave workflow instance terminates here.

5. Conclusion

Many related research works have focused on the agent-assisted conceptual models of B2B or B2C workflows, however, the primitive agent model, operational behavior and collaborations are seldom detailed. In this paper, we aim at designing an *agent-based* system architecture for primitive workflows. We model the business process scenarios as the interactions among the system components, users, and software agents, and exploit the design from the agent's viewpoints to cope with the inappropriateness appeared in traditional WfMSs.

Besides, the system components and software agents with their contents and capabilities are all clearly illustrated. The interactions among them to fulfill the typical workflow behaviors are also shown with sequence diagrams. In the future, we plan to investigate the design of the workflow definition tools from the agent's viewpoints, since our goal is to utilize the properties and capabilities of software agents to enhance the development process of workflow-based applications. In addition, we are looking at the structure of the client tool for introducing the concept of active agents for better interface representations and manipulations.

References

[1] Gregory Alan Bolcer and Richard N. Taylor, "Advanced Workflow Management Technologies", Software Process: Improvement and Practice, Volume 4, Number 3, pp.125-171, September 1998.

[2] [URL] The Workflow Management Coalition, http://www.wfmc.org

[3] Nicholas R. Jennings, Timothy J. Norman and Peyman Faratin, "ADEPT: an agent-based approach to business process management", ACM SIGMOD Record, Vol 27, Issue 4, pp.32-39, ACM Press, New York, Dec. 1998.

[4] Liangzhao Zeng, Ngu Anne, Benatallah Boualem and O'Dell, Milton, "An agent-based approach for supporting cross-enterprise workflows", 12th Australasian Database Conference, pp.123-130, Gold Coast, Queensland, Australia, 29 Jan – 2 Feb 2001.

[5] Chen, M. –F., Liang, B. –S., Lin, J. –R., and Wang, F. –J., "Enacting a Software Development Process," IEEE ICSCCS'97, pp. 3-12.

[6] Chen, M. –F., Liang, B. –S., and Wang, F. –J., "A Process Centered Software Engineering Environment with Network Centric Computing," IEEE FTDCS'97, pp.234-239

[7] [URL] Flowring Corp., http://www.flowring.com

[8] The Workflow Management Coalition, "Workflow Management Coalition Workflow Standard Workflow Process Definition Interface – XML Process Definition Language", Document Number WFMC-TC-1025, Oct 2002.