Agent-based Approach to Workflow and Supply Chain Management

YUNHE LIAO, HINNY P. KONG
School of Electrical and Electronic Engineering
Nanyang Technological University
Block S1, Nanyang Avenue
Singapore 639798

Abstract: Enabled by the evolution in information technology, companies are constantly trying to re-engineer their business processes to be more productive and efficient to respond to the quickly changing environment. A number of workflow management systems (WFMSs) have been developed by various research groups and companies to meet these requirements. After decades of evolution, they are moving from centralized to distributed mode and have become more and more flexible. However, in executing processes of different enterprises, flexibility and distributed processing capability are still the main challenges. In this paper, we propose an agent-based WFMS (aWFMS), which is designed to be flexible, distributed, easily integratable, and capable of executing processes enterprise-wide. XML is chosen as our agent communication and process definition language. The proposed model is applied to Supply Chain Management (SCM) applications.

Keywords: Workflow, Workflow Management System, Agent, XML, Supply Chain Management

1. Introduction

Workflow as a concept has been existed for decades. The earliest known WFMS is Officetalk developed by Ellis and Nutt [1] in the Xerox PARC. In the 1980s, some office automation and document imaging workflow product have been developed [2]. Since the early 1990s, workflow has been the focus of intense activity in terms of products, standards and research work [3]. There are various definitions been proposed for Workflow and WFMS over the years. In 1993, some workflow vendors, Analysts and Research Groups came to form the Workflow Management Coalition (WfMC) [4]. WfMC now is the major group that involves in workflow management standardization.

Despite the potential benefits that workflow technology can bring to companies and the large number of WFMSs available in the market, Amit and his colleagues from Georgia [5] observed that current WFMSs were not quite successful. And they concluded that lack of flexibility, real standard, and restricted functionality are the main reasons that preventing companies from investing in workflow. Another interesting phenomenon is that despite the large number of technical users who need to customize their own workflow solutions, most existing WFMSs target only at non-technical users. Besides workflow functionality they also provide high level visual tools that support workflow configuration, but are close-ended system that are not extendable or programmable [6]. The functionality gap between the current WFMSs and requirements of inter-enterprise applications and technical users suggests the future direction of research for workflow management.

In this paper, we propose an aWFMS model based on IBM’s Aglet Software Development Kit (SDK) (http://www.trl.ibm.com/aglets/). The proposed aWFMS consists of various types of agents. We define the abstract class for each type of agent. New agents can be built by developers and plugged into the aWFMS to support various applications. By using component agents to build an aWFMS, great flexibility and integrability can be achieved. Workflow
process can be executed distributively over a few aWFMSs that cooperate with each other using agents. XML is chosen as our agent communication language.

The rest of this paper covers a review of some of the better known WFMSs (Sect. 2), IBM’s ASDK and XML technologies (Sect. 3), our proposed WFMS model (Sect. 4), and demonstration of this model in SCM applications (Sect. 5), and opportunities for further work (Sect. 6).

2. Review of Existing WFMSs

A good number of WFMSs were offered in the market in the 1990s, such as I-Flow from the Fujitsu Software Corporation [7], Emailflow for Exchange/SMTP from Docman [8], EntireX from Software AG [9], Exitoca/FMQM from IBM [10], FloWare from Plexus [11], JetForm Workflow from Jet Form [12], Livellink Workflow from the Open Text Corporation [13], Mentor-lite from University of Saarland [14], WIDE from the WIDE research group [15], Workflow Suite from the Ultim us Corporation [16]. The number is estimated to be between 200-300 in 1999 [5]. Herein, four of the WFMSs that have deeper impacts on workflow technology are reviewed: Exitoca/FMQM, Mentor-lite, WIDE and EntireX.

- Exitoca/FMQM From IBM
  FlowMark is the initial workflow product from IBM [9]. It is built based on the client/server architecture. The server, in turn is a client of a centralized database, ObjectStore, where both runtime and buildtime workflow information is stored. At runtime, information flows from the database to the server, from the server to the runtime client, and vice versa. The centralization of database forces the clients to be connected to the server at all times to be able to progress in their work, thus it is vulnerable to server failures and offers limited scalability due to the potential performance bottleneck caused by the centralized server. To overcome these problems, IBM designed a distributed architecture for a WFMS with improved reliability and scalability.

The new system is named Exotica/FMQM. In the new system, execution of business process is distributed. Each node of the process maintains the required persistent storage.

This information is used to recover from failures occurred at this node. A messaging system called MQSeries is used in Exotica/FMQM to support asynchronous communication between different parties of the WFMS.

- The Mentor-lite WFMS
  The Mentor project (Middle ware for Enterprise wide Workflow Management) is a collaboration project of the University of Saarland and the Union Bank of Switzerland. The early version of workflow management system models workflow with state and activity charts [14]. Initially it focused on deriving distributed workflows starting from formal specifications. An algorithm that transforms a centralized state and activity chart into a provably equivalent partitioned one and is suitable for distributed execution was developed. As work went on, the researches realized the limitations of the workflow architectures.

Consequently they continue to work on Mentor-lite, a second-generation Mentor system. Mentor-lite is a lightweight WFMS. It separates the workflow kernel functionality from the additional functionality of workflow systems, and it supports the incremental integration of workflow technology within existing business environments. System components like history management and worklist management are extensions of the kernel. An invocation interface for application programs is provided by a generic IDL interface on the engine side and specific wrappers on the application side.

The workflow engine kernel consists of three components: (i) the workflow specifications interpreter, (ii) the communication manager (ComMgr), and (iii) the log manager (LogMgr), which are closely integrated with the interpreter. They use state and activity charts for process definition. A workflow process can be executed in a distributed manner over several workflow engines at
different sites. A separate workflow log is used at each site where a Mentor-lite workflow engine is running. And a few workflow engines at different sites can share the same worklist database for workflow process definition (workflow repository).

- **The WIDE WFMS**
  WIDE is an European Esprit project involving Sema Group, Politecnico di Milano, University of Twente, ING Bank and the Manresa Hospital[15]. This project has focused on extending distributed and active database technologies to the workflow arena. The WFMS system architecture consists of three layers, the workflow server layer, basic access layer and the database layer. The workflow engine provides functionalities to run locally the tasks and their activation according to the Workflow schemas stored in it, including exception support. It also interacts with other servers to perform tasks like task distribution, remote execution and global transaction management. While Workflow specific data and application data are provided by a database management system. The interaction between the workflow engine and the database is through the basic access layer module.

- **EntireX from Software AG**
  EntireX from Software AG is an XML-based workflow product. It claims to have over 1000 customers by the end of year 2001 [9]. And it consists of three components, EntireX Orchestrator, EntireX Communicator, and EntireX Adapters. The Orchestrator allows systems to read and interpret data in different database (Oracle, SQL). Workflow user can perform real-time data transformation by graphical point-and-point click approach without coding. EntireX Communicator provides support for simultaneous request and reply messages, it also manages interactions between XML-enabled and non-XML-enabled systems, EntireX Adapters are pre-build adapters that enable software integration with standard applications such as Siebel, SAP and Vantive. It also can integrate with various databases such as Oracle, Adabas; and other technologies such as CICS and MQSeries.

The four WFMSs reviewed in this Section have successfully separated application logic and process logic, and they evolved to support distribution execution of processes. They have different emphasizes. Exotica/FMQM has focused on using persistent messages for communication between distributed workflow engines. In the Mentor-lite, attention is concentrated on separating the workflow kernel functionality from the additional functionality typical of workflow systems, thus supports the incremental integration of workflow technology within existing business environments. The WIDE has focused on extending distributed and active database technologies to the workflow product. EntireX is XML-based, and can integrate different databases and applications residing in different platforms. EntireX also provides graphical user interface to enable workflow user to define processes.

However, none of them has illustrated how to solve the cross-enterprise boundary problem. It is difficult for other WFMSs to communicate with them, particularly because of the different message formats used. And they are initially designed for certain applications, thus the functionalities are quite restricted. Our solution is presented in Section 4.

3. Review on IBM’s Aglet SDK and
XML Technology

A software agent is a program that works on behalf of a human user. A mobile agent has the added ability to travel autonomously from machine to machine on a network. An agent usually exhibits the characteristics of autonomy, social ability and reactivity [17]. Since workflow is concerned with the automation of procedures where documents, information or tasks are passed between participants, using of agents especially mobile agents in WFMS will greatly enhance its flexibility, efficiency and scalability.

IBM Aglet SDK was chosen based on the strength of its features and our in-house experiences. Aglet SDK is a Java-based framework for implementing mobile agents. The SDK provides a comprehensive object-oriented programming interface for developers. Process state is maintained when mobile agent travels from one site to another. An aglet can be created within the specified context from the specified class, it can be cloned, dispatched to another host or be retracted from there. It also can be de-activated and temporally stored into disk and activated again into the same context when needed. It uses the delegation event model introduced in the JDK, and it adds three event listeners, which are: clone listener, mobility listener and persistent listener to support various operations. Aglets can communicate with each other via message passing. MAgNET [18] is an e-commerce product built with ASDK. Their experiment results have demonstrated the feasibility and flexibility of Aglet SDK.

XML is an ideal language for agent communication and process definition for its open standard. XML document is a truly simple, flexible and open architecture that allows all applications on various machines on different platforms to interact. It is quickly implemented by major vendors such as IBM, SUN and Microsof t since its release. Now nearly all commercial DBMSs have been extended to handle XML documents [19]. For its open standard, XML is believed to be able to eliminate the need for customer interface with every customer and supplier, allowing buyers to compare products easily across many vendors [20]. Thus, in our aWFMS, mobile agents carry XML-formatted messages to communicate to other agents residing in other aWFMSs. XML is also chosen as our process definition language, which is conforming to the XML process definition interface proposed by WfMC [21].

4. The Agent-based WFMS (aWFMS) model

The proposed aWFMS model (Figure 1) consists of four types of stationary agents and one type of mobile agent. All of them are implemented using the base agent class defined in the aglet class library. The stationary agents are: Workflow Control Agent, Process Definition Agent, Role Agent, and Administration Agent. Process execution is coordinated by various types of agents interacting with one another through message passing. Detailed description of these agents is presented below:

- Workflow Control Agent - This agent plays a very important role in an aWFMS. It is the ‘brain’ of a workflow engine. It interprets the process definition at run time, observes the execution state of process, checks the transition conditions and assigns roles to the respective parties. It controls the instantiation of all other agents and applications.
• Process Definition Agent – This agent helps to translate a business process from real world into a formal, computer processable definition, which can be interpreted by the workflow control agent at run time.

• Role Agent – This agent is instantiated by the workflow control agent to perform various roles, such as analyzing data and performing calculations. Once activated, a role agent can keep listening to wait for next instruction from the workflow control agent, it can also check the worklist maintained by the workflow control agent regularly, pulling jobs from the worklist.

• Administration Agent - This agent allows the administrator to add/modify parameters of the aWFMS. It presents a GUI to allow administrator(s) to perform functions such as user management, role management and resource management.

• Mobile Agent - This is a special agent that can move from host to host under its own control, it helps realize the distribution execution of workflow process. A mobile agent can be used to perform the role of data collection from many places, data searching and filtering, dynamic data monitoring, targeted information dissemination, negotiating, and parallel processing [22].

Each type of agent has its own database to support its operation (Figure 1). Agents in the aWFMS can communicate by using mobile agents, remote messages as well as local messages (Figure 2). Local messaging can pass any kinds of objects as arguments, while remote messaging and message carried by mobile agents that travel on the network can only be of Java type that implements the java.io.Serializable interface [23]. XML is chosen as the agent communication language, Document Object Model (DOM) as defined by W3C is used to represent a XML message and JAXP from SUN is chosen as the parser. However a DOM object itself is not serializable, it must go through a process namely object-serialization before transfer. At the receiver side, the serialized XML message is converted back into an XML document object to be handled by the agents.

In our aWFMS model, the process definition agent helps to translate a problem to a computer processable form, ensuring the separation of process logic from application logic, thus the proposed is quite flexible. Workflow activities can be performed using various role agents which can be easily built from the abstract class we defined, ensuring the aWFMS to be scalable and integrable. Mobile agent is used to carry data, process information to talk to agents in other aWFMSs, and open standard language XML is chosen as the agent communication and process definition language, making it possible to execute processes across enterprise boundaries.

Security is essential to any WFMSs. As our aWFMS consists of various types of agents, the security issue of the aWFMS becomes the issue of agents. In our system, domain authentication and agent authorization are used to prevent unknown agents from attacking.
the system.

5. A Supply-Chain Workflow Management System

The aWFMS model proposed in previous section is applied to Supply Chain Management (SCM). There are two predominant reasons for choosing SCM to demonstrate: (1) SCM presents significant challenges in executing processes across enterprise boundaries [5]; (2) SCM plays a vital role in a modern enterprise. According a recent survey [24], supply-chain problems cost companies between 9 and 20 percent of their value. With the overall B2B market estimated to reach $7 trillion in 2004 [25], it is obviously that effective SCM solutions could save companies billions of dollars and also make them more competitive in the new economic environment [26].

A SCM system usually consists of the following parties: the buyer who initiates the order by specifying the purchasing criteria, the supplier who analyses customer’s orders, verifies product storage and schedules the delivery, and the transport (logistics) provider who distributes the goods. For the purpose of demonstration, our focus is on the order fulfillment process. Our system will handle this process as shown in Figure 3. The detailed operations involved in this process are:

1. A purchasing request is entered through an interface provided by the buyer aWFMS users through HTTP, WAP, or GUI as provided by the workflow control agent.
2. The request is approved by a person in charge.
3. A querying agent is dispatched with the buying criteria to the destination(s) specified.
4. A supplier aWFMS receives a querying agent and provides the requested information.
5. Querying agent may continue to travel to the next site and send a remote message to the buyer aWFMS to report the status.
6. Once the querying agent gets the information it wants, or when all sites have been visited, it will return to the buyer aWFMS. Meanwhile, if there is an offer found to be good enough, it will make a reservation.
7. Buyer’s workflow control agent initializes a role agent to analyze all the information the querying agent collects and issues an order.
8. A supplier aWFMSs then checks the inventory level and sends a querying agent to distributors’ sites to arrange the delivery schedule.
9. Now steps 4-6 are repeated between the suppliers and their respective transport (logistics) providers.
10. A supplier then places an order for transportation and confirms the purchasing order from the buyer.
11. The buyer makes initial payment to the supplier.
12. The supplier makes initial payment to the transport (logistics) provider.
13. The transport (logistics) provider confirms the delivery order with the supplier.
14. The supplier confirms the purchasing order with the buyer.
15. The buyer presents the result to the purchaser, informs the respective parties, and records the purchasing order in its database.
16. Now the three parties wait for the scheduled time to execute the order.

As each party in the supply chain has its own process definition, the supplier itself can be a buyer and vice versa. A multi-level supply chain process can be easily realized by using our aWFMS (Fig. 4).

6. Summary and Future Research

Workflow technology remains an area that receives much attention and evolving very fast. Observing that existing WFMSs are not quite successful in the market, particularly due to lack of flexibility, distributed processing capability, and interoperability, an agent-based WFMS (aWFMS) model is hereby
proposed in this paper. The proposed model has demonstrated great flexibility, distributed processing capability and intergrativity. Agent and XML technologies enable the system to execute processes enterprise-wide. The aWFMS model was demonstrated successfully in the SCM application. Next, it will be applied to project management.

Future research will focus on enhancing the security of the proposed model, e.g., encrypt the data carried by mobile agents. For SCM applications, there is a need to study how negotiation should be carried out between the buyer and suppliers using mobile agents.

References: