Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT•

Information and Software Technology 46 (2004) 373–382

INFORMATION AND SOFTWARE TECHNOLOGY
www.elsevier.com/locate/infsof

A document-driven agent-based approach for business processes management[☆]

Jong-Yih Kuo

Department of Computer Science and Information Engineering, Fu Jen Catholic University, HsinChuang, Taipei 242, Taiwan

Received 14 October 2002; revised 4 February 2003; accepted 17 August 2003

Abstract

Due to the development of Internet and the desire of almost all departments of business organizations to be interconnected and to make data accessible at any time and any place, more and more workflow management systems are applied to business process management. In this paper, a mobile, intelligent and document-driven agent framework is proposed to model business process management system. Each mobile agent encapsulates a single document, which includes a set of business logic. It can achieve (1) trace ability: a function that enables administrators to monitor document processes easily, (2) document life cycle: a feature using agent life cycle to manage document life cycle and concurrent processing, and (3) dynamic scheduling: a document agent can dynamically schedule its itinerary, and a document control agent can dynamically schedule its services. We also implemented an official document management system explaining our approach by Aglets.

Keywords: Mobile agent; Business process management; Document-driven approach

1. Introduction

Workflow management promises a new solution to a traditional problem: controlling, monitoring, optimizing and supporting business processes. Traditional business process management system utilizes distributed objects management technology [21] to manage and negotiate the heterogeneous and distributed business activity of enterprises. Currently, existing enterprises are expanding from local organizations to international distributed companies where business processes dynamically form temporary alliances, joining their business in order to share their costs, skills and resources in supporting certain activities [7, 30]. Many researches proposed intelligence agent technology to automate process operation integrating the different business systems to reduce the complexity [14,15,22]. Several benefits can be obtained by means of agent technology in the business process management [13], including autonomy, social ability, pro-activeness, and responsiveness. Agents perform the majority of their problem solving tasks without the direct intervention of humans or other agents, and they have control over their own actions and their own internal state. Agents interact, when they deem appropriate, with other artificial agents and humans in order to complete their problem solving and to help others with their activities. Agents take the initiative where appropriate and perceive their environment and respond in a timely fashion to changes that occur in it.

Moreover, mobile agent technology offers several advantages over traditional approaches to Internet applications [3,25,24]. It can proceed without continuous network connections to save significant bandwidth by moving locally to the resources they need, because interacting entities can be moved to the same site when connections are available and then interact without requiring further network connections. The mobile agent can carry the code to manage remote resources and do not need the remote availability of a specific server, so it has intelligent information gathering to work with mobile computing systems.

As complex business processes rely on intensive information exchange with the company's environment, they are document-driven by nature [1]: employees deal with and react to information and knowledge transferred by

[☆] This research is partially supported by the National Science Council under grant NSC 91-2213-E-030-010. We thank Prof. Jonathan Lee for his fruitful discussion on an early draft of this article.

E-mail address: jykuo@csie.fju.edu.tw (J.-Y. Kuo).

and embedded in all kinds of documents, including forms, letters, books, manuals, records, either electronic or paper-based. Documents reflect many results of business processes in the private as well as in the public sector [31]. Most of business processes are centered about the 'disposal record', a document that allows a company to dispose of a certain type product or process in the approved way. Consequently, one would like the business process management system to automatically offer access to relevant knowledge source, or to even directly 'pump' information items extracted from incoming documents to the appropriate places in the data models of the actual workflow instance.

In this sense, we attempt to integrate mobile agent technology and document-center concept [1,31] to provide flexible business process management system. In this paper, we base on our previous works [18,19] to propose a document-driven agent-based (DAB) framework dealing with dynamic workflow management. The business documents can be packaged and handled by mobile agent over distributed business environment. We also use Aglets [20] to implement the business process management system for our case study explaining our approach. In the sequel, we first review the related researches. The DAB approach is fully discussed in Section 3. The case study is outlined in Section 4. Finally we conclude the paper by outlining the potential benefits of the proposed approach.

2. Related works

Works in a number of fields have made its mark on our research. Technologies for business process management systems are evolving across many different dimensions namely, Object-orientation, workflow management, intelligent agents and Internet. Our approach has drawn upon several ideas from the Object-orientation, mobile agent technology, document-center concepts, and techniques in handling workflow management.

2.1. Object-oriented workflow management

Object orientation represents an ideal technology for designing and building workflow management systems. Flexibility is facilitated by the use of polymorphism, inheritance and encapsulation. A workflow management system has an open architecture which can be specialized and extended through inheriting from an object-oriented component library.

Fakas et al. [5] design the workflow intelligent business objects (WIBO) to model and implement workflow management. They use WIBOs to build workflow system that exhibit intelligence, autonomy, collaboration and co-operation. WIBOs also exhibit typical characteristics of object oriented system, including reusability, inheritance, encapsulation, and composite workflow. The use of WIBOs can result in a better dynamic management of the process and utilization of resources. The proposed architecture cannot support the interoperability between WIBOs and other groupware and workflow systems.

Distributed object management (DOM) [21] supports the interoperability and integration of heterogeneous distributed systems and applications implementing business processes. DOM allows workflow management systems to cope with replacement, migration, and evolution of heterogeneous distributed systems or changes in their functionality and data. A decentralized workflow framework, call EVE [9], uses events and Event-Condition-Action (ECA) rules as the fundamental concept for defining and enforcing workflow logic. It supports event based distributed workflow execution. Workflow specifications are mapped to the software architecture model where the workflow tasks are modeled as reactive software components, which react to specific events in a well-defined manner. Ho et al. [11] present a collaborative word processing system to allow multiple co-authors editing a consistent version of the document. The system is built on a CORBA-based workflow framework in which the flow of the document components among the co-authors is governed by a workflow schedule that can be adapted dynamically during execution.

2.2. Document-oriented business process management

The genesis of workflow management software was probably in automating document-driven business processes [27]. Some of the early products were extensions to the document imaging and management software. The workflow management systems for office automation can support document management imaging, application launching, coordination, collaboration, and co-decision [8]. At present, the systems must support enterprise-wide workflow applications effectively.

For flexible control and data flow modeling, Joeris introduced an approach to support cooperation on the workflow level which takes versioning and different forms of data interchange between activities into account [16]. They concentrate on both document level and workflow level within a comprehensive process management system. The backbone of the integration of tasks and documents is the input/output relationship which is defined by the parameter specification of a task.

The approval of hazardous-waste disposal is a highly regulated, interorganizational business process. Wewers et al. present a system that supports this process and embed it in a framework for interorganizational, document-oriented workflow [31]. Due to the document-orientation of the disposal record process, it can suitably be supported by workflow management systems that are based on document management system.

In Ref. [30], Aalst describes a Petri nets technique to model and analyze a workflow process. The generation of a workflow process is based on a bill-of-materials (BOM).

A BOM specifies which materials are needed to manufacture a product. The well-known BOM document can be used to describe the product that is manufactured using a workflow management system. It allows workflow designers to think in terms of the end-product instead of the internal process and constitute a basis for the automatic configuration of a workflow management system on the basis of a BOM. The BOM provides product-centric view and the Petri-net provides a process-centric view.

2.3. Agent-based business process management

Usually, agents in workflow management systems act as personal assistants, performing actions on behalf of the workflow participants, continuing watching for information and responding to it when it meets certain specified criteria. In multi-agent system, agents have control over the tasks that they may perform, the resources available to them and how they coordinate their activities with other agents.

A 'WorkWeb System' [29] is an expanded workflow system, that is able to management, control and coordinate office resources, especially human resources. A business process tactics (BPT) agent in the system autonomously manages each workflow process instance trying to acquire the necessary resources to complete it in time. The system also provides interface to manage office goals and several workflow re-planning algorithms to handle exceptional cases.

In Ref. [10], Gou et al. treat agents as autonomous entities with abilities to solve problems independently. They propose an agent-based approach for workflow management, aiming at achieving flexible and dynamic workflow management in the distributed environment. In their agent hierarchy, there are three kinds of agents at three neighboring levels: activity agents, role agents and actor agents. For organizing and collaborating the three kinds of agents at three levels, they provide two collaboration patterns inside the agent hierarchy: activity-role-actor assignment and feedback patterns. The proposed agent model consists of three sub-agents: a message-receiving sub-agent, a decision-making sub-agent, and a message-sending, sub-agent. Connected by two message queues, these three collaboratively operating sub-agents can complete functions of an agent effectively.

Advanced Decision Environment for Process Tasks (ADEPT) [13,15] adapts multi-agent architecture composed of a number of autonomous agencies to provide a well structure for dealing with business process management. An agency contains a single responsible agent, a possibly empty set of subsidiary agencies and a set of tasks that are under the direct management of the responsible agent. All ADEPT agents have the same basic internal architecture, illustrated by the responsible agent of agency. An agent has six functional components: communication module (CM), interaction management module, situation assessment module, service execution module (SEM), acquaintance model, and self-model. Although ADEPT has been attracting widespread interest in business process management, several developmental obstacles remain. All of agents are stationary in the environment, so the major weakness of this approach is the lack of monitoring, trace ability, and mobility for business activity.

The existing approaches lack the integration of the document-oriented concept and a development framework of mobile agent system from requirements analysis to implementation. In this paper, we attempt to integrate mobile agent technology and document-center concept to propose a DAB framework for flexible business process management system.

3. Document-driven agent-based (DAB) framework

We proposed the DAB, a document-driven approach that applied the intelligence agent and mobility technology to encapsulate a document as a mobile agent that handles the business activity. The mobile agent can trace and monitor the business document, so that the business process may be better managed.

3.1. The internal architecture

All DAB agents have the same internal architecture (Fig. 1). This involves an agent, which is responsible for managing the agent's activities and interacting with peers and agency. The agent has a number of functional components responsible for each of its main activities. The CM routes messages between an agent and both its agency and peers. The mobility module (MM) is designed for the mobility ability of agent, including the mechanism of dispatch, retract, dispose, clone, activate, deactivate, and create. The SEM is responsible for managing services throughout their execution. The knowledge-base handler (KBH) is a knowledge storage for the reasoning mechanism where the agent is committed, describing the services the agent can provided, running time application/service specific information and generic domain information. The internal architecture is designed to ensure maximum flexibility to adopt as a business process changes.

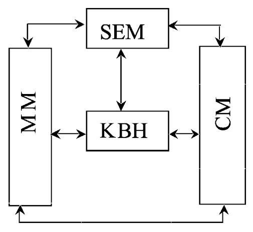

Fig. 1. The internal architecture of DAB framework.

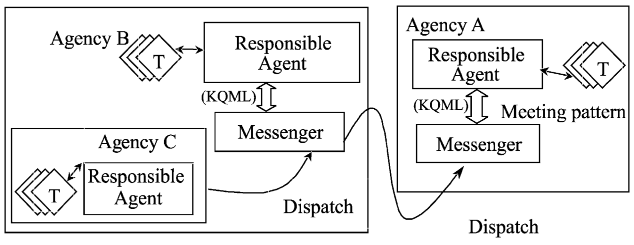

Fig. 2. The external architecture of DAB framework.

3.2. The external architecture

The external architecture represents the structure of the multi-agent system as a whole, and the role of an agent within that system. The architecture is broadly based on the ADEPT model. The architecture is composed of a number of autonomous agencies (see Fig. 2). Agency A and B are peers, and agency C is the subsidiary of agency B. The responsible agent of agency C generates a messenger agent that moves locally to agency B in order to interact with each other. The architecture can model the organizations of enterprises, such as structure of hierarchical or flat organizations, or a mixture of the two, through the concepts of agents and agencies. It also provides a design and implement pattern for multi-agent business process management systems.

In the proposed architecture, two agent types are provided: the responsible agent and the messenger agent. The responsible agent is a stationary agent that creates and dispatch the messenger agent into the right server for some missions. The user can control it through the user interface. The messenger agent is a mobile agent that can pack a document up and run the business process described in the document. The concepts between mobile agent and stationary agent differ in terms of the communication message, the life cycle of agent, and the cognition of context. The comparison between the two types of the agents is summarized in Table 1.

Table 1
Responsible agent vs messenger agent

	Responsible agent	Messenger agent
GUI	Yes	No
Communication	User, messenger	Responsible, messenger agent
Service execution	Maintain services, schedule services	Maintain documents, schedule ininerary
Mobility	No	Yes
Knowledge-based handler	Acquaintance	Agent locker pattern

3.3. The agent communication and negotiation

Autonomous agents have a high degree of self-determination: they decide for themselves what, when and under what conditions their actions should be performed. These agent need to interact with other agents to achieve their objectives. The objectives of these interactions are to make other agents undertake a particular course of action (e.g. perform a particular service), modify a planned course of action (e.g. delay or bring forward a particular action so that there is no longer a conflict), or come to an agreement on a common course of action.

In order to communicate with remote agents, the responsible agent created a messenger agent that moves itself to the remote content. For the communication and negotiation between the different types of agent or in the different agencies described above, we have chosen for the Knowledge Query and Manipulation Language (the KQML) [6]. A message is created with structured layer of KQML for communication information and Knowledge Interchange Format (KIF) for the main information-parameter, tag, etc. In order to realize the communication among the agents, we apply the Meeting design pattern [2] to define how synchronization between different agents that wish to interact with each other is to take place. Once an agent (which wishes to interact) arrives at the particular agent service identified, it needs to register itself with a manager object (such as the responsible agent), and notify other agents of its presence. The participants involved in the Meeting design pattern include the Responsible agent, Meeting object, and the Messenger agent. The Responsible agent is responsible for coordinating interaction between the incoming agents, and informing them of the arrival or departure of agents. The Meeting object stores the address of the meeting point and a unique meeting identifier.

The negotiation model in our framework is based on a variation of the two parties, many issues value scoring system presented in Ref. [26]. Our variation transforms that model into a many parties, many issues model. Our model of multi-agent negotiations is based on a set of manually influencing two-parties, many issues negotiations. The model has the following negotiation characteristics. (1) A given service can be provided by more than one agent. (2) Individual agent can be both clients and servers for different services in different negotiation contexts. (3) Negotiations can range over a number of quantitative and qualitative.

3.4. Agent-based document-driven approach

Most workflow management systems follow an activity-centered approach and have several limitations with respect to support for cooperative work and data-centered processes. Further, they lack the ability of dynamic workflow modifications and flexible control and data flow modeling.

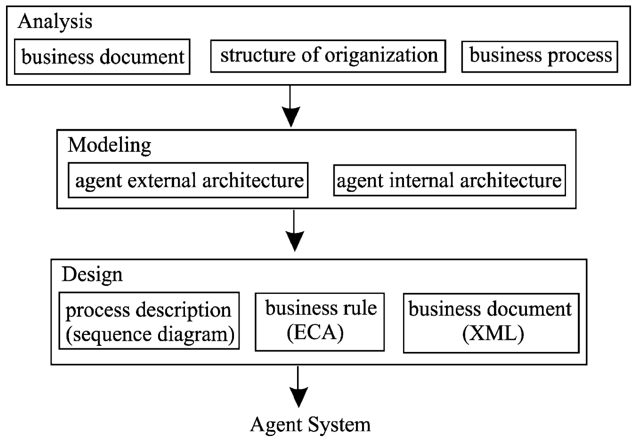
Analysis
business document
structure of origanization
business process
Modeling
agent external architecture
agent internal architecture
Design
process description (sequence diagram)
business rule (ECA)
business document (XML)
Agent System

Fig. 3. The framework of agent-based document-driven approach.

Document-driven aspects are not only covered by cooperation support. Rather, we have to take document-driven processes like object life cycle models or object migration models into account. Furthermore, the document structure and dependencies may determine task decomposition and ordering. The evolution of the document content or structure and dependencies will cause evolution of the process instance, which cannot be defined a priori. As suggested in Fig. 3, we advocate an integrated approach to designing a DAB business process management system as following three phases.

Analysis phase. In this phase, we analyze the organizations of enterprise, the business documents and related activities from various industrial and commercial aspects. The output is the use case model. The steps of analysis are described as follow:

- *To extract the key business documents*: First of all, the key business documents that the involved organizations execute together. In general, business documents reflect many results of business processes in the intra-organization as well as inter-organizations.
- *To identity the structures of organizations and the roles of divisions*: For each key document, we view the related organization as a collection of roles, that stands in certain relationships to one another, and an division as an actor, that deals with business activities based on the business document.
- *To abstract the common the characteristic of the business processes*: The business processes are defined across different organizations that serve to complete a task which is the goal of a business document.

Modeling phase. In this phase, the external and internal architecture must be built. Both the architectures models can be first described in their high-level abstract forms, which can be further refined into more detailed specifications in the next phase.

- *The external architecture*: Each organization may have many divisions, and then we construct one agency and some subsidiary agencies in the system. To define an agency, it suffices to define the responsible agents of divisions in the organization, how these agents relate to one another, and how an agent can interact with other agents.
- *The internal architecture*: A business document that includes business logic can be packaged and handled by a messenger agent over distributed business environment. The messenger agent also manages the life cycle of the document. The responsible agent is a stationary agent that creates and dispatch the messenger agent into the right server for some business activities. So the messenger agent has the MM, and the responsible agent has the agent management module.

Design phase. The business logic comprises process descriptions, business rules, and the document related information. The dynamic properties of the agent system are characterized by the following.

- *To specify the process descriptions*: For execution of the business activities of document, the business document is routed from one employee of division to the other. A sequence diagram shows the sequence of messages exchanged of the business document by the set of agents performing a certain business process. For business automation, we then formally specify the routing sequence using task state expressions [33].
- To formulize the business rules: ECA rules [4] automatically perform actions in response to events provided stated conditions hold. An event can be message from an agent (e.g. a new document arrival), a property change (e.g. updating document) or a state transition (e.g. document rejected). Conditions are predicates that are used to decide what actions should be executed and are applied to the event or events that precede it and return true or false (e.g. trade price is within acceptable limits). Actions represent things to do, and are typically built-in functions or any business strategy.
- *To represent the business document*: We use XML to represent the business documents and business logic. We also use XSLT [32] to control the access rights of the business documents.

Finally, we can use an agent development kit to implement the agent system (e.g. Aglets [20], Voyager [23] or JATLite [12]).

4. A case study: official missives management

Currently, existing universities are expanding from local organizations to international distributed colleges where business process dynamically form temporary alliances, joining their business in order to share their costs, technologies and resources in supporting certain activities. So that, they must use a office missives management system in the Internet environment. The mobile agent system is suitable alternative to operate in the internet which is probably based on the unreliability and low bandwidth net.

This section briefly illustrates how the method can be applied, through a case study of the analysis and design of a DAB system for managing an official missive process. First, we use the use case diagram and the activity diagram to analyze the management system. Second, once the system architecture is established, the internal and external architectures of agent system are designed to fit the management system. Finally, the official missives management system is implemented by Aglet [20].

4.1. The requirement analysis of document-driven agent system

The process is initiated by an office clerk of a department at the university. The clerk writes a missive and sends it to the chairman of department gives some comments and then sends the missive to the chairman of the college. The chairman of the college gives some more comments and then sends the missive to the office of Academic Affairs, Student Affairs, General Affairs, Accounting, and Personnel. The directors of these, offices give their comments, and then send the missive to the secretariat As soon as the head of secretariat receives the missive, she/he sends it to the president of school. Finally, the missive is sent back to the department clerk.

We identify two actors: clerk and chairman of division. The use case diagram is shown as Fig. 4. Then the scenario of the agent-based management system is shown as follow:

- Each user of department of college first activates their document agent system, and then the agent system triggers the web browser. The user can manage their document (messenger) agent through the web browser.
- When the user wants to write a missive, he will have to choose the type of missive and the degree of emergency. The ECA Parser will check the business logic of the missive and the roles participated this process.
- If there is any error occurred in the process of parser, the system will demand the user to make some specific modification. Otherwise, the document control agent will create a document (messenger) agent to finish the process of missive.
- The document agent carries the missive to the destination office and queues the schedule along with other missives from another department. In the missive is an emergency missive, the document agent will create a new window and notify the user to deal with the missive in time.
- The missive sender can trace where the missive is and what the situations are. The situations of missive contain the arrival time, the finish time, the waiting time, and any exception event.
- The missive receiver can click the review function. If there are too many unread missives coming in to the receiver, the document agent will give the receiver a warning. If the receiver completes the review, the agent system will continue to route the missive based on the business logic and the knowledge of agent.
- If the user has any problem, he can send message to the system by message agent.
- Finally, the document agent takes the missive and the results back to the original sender.

Finally, We use the activity diagram to model the system behavior as Fig. 5.

4.2. The architecture design of the agent system

The official document agent system is a web-based system. The system architecture is shown as Fig. 6. Each client and local server of department must install the Aglets environment that is viewed as an agency of the agent system. Each agency encompasses the document control agent, the document agent, the message agent and the, KBH. Every server must also install a web server. Each user of department first activates their message agent, and then the agent can trigger the web browser. The web browser can

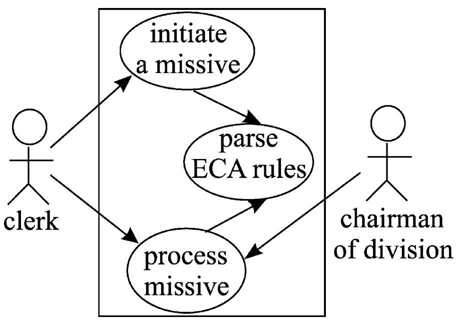

Fig. 4. The use case diagram of official missives management system.

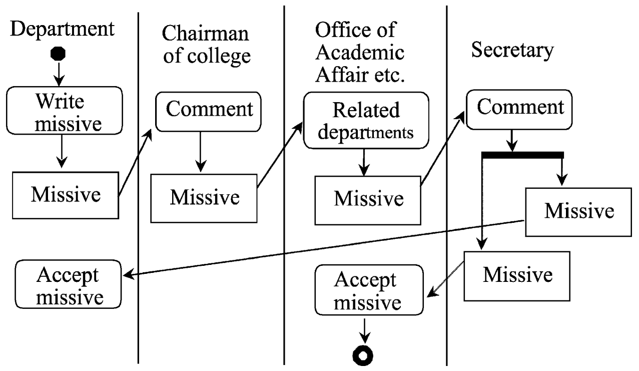

Fig. 5. The activity diagram of official missives management system.

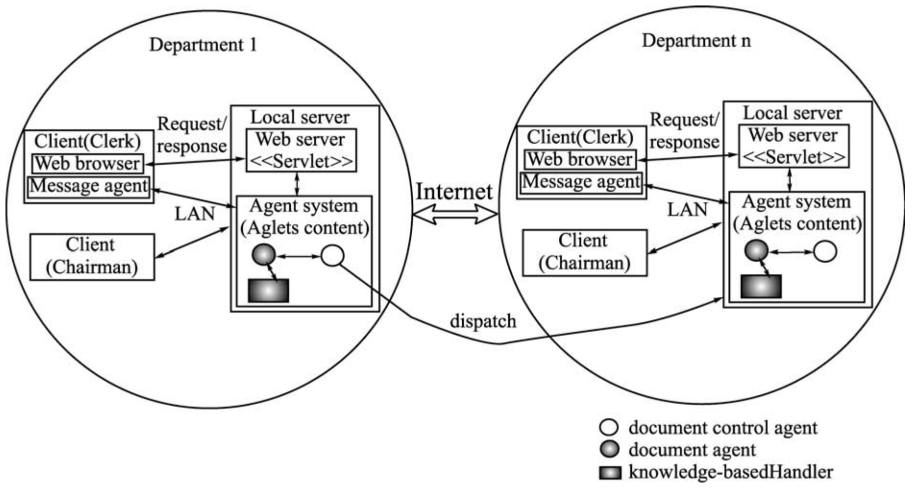

Fig. 6. External architecture of agent system.

connect to the local server of the department through Http request.

The functionalities of agent system in the client side are described as follow:

- The agent system of the client side can activate the web browser and communicate with the document control agent in local server.
- The clerk can use web browser to connect into local server, that he can monitor the status of his missive.
- The message agent can listen to the messages form agent system of local server. If any missive arrives, the agent will notify the clerk or chairman.

The functionalities of local server are explained as follow:

- If a clerk submits his missive to the local web server, the web server may transfer the data to document control agent of agent system.
- The document control agent then creates and dispatches a document agent (mobile agent) to all target departments or divisions. The document control agent extending the responsible agent is responsible for managing the document agent and the message agent. The document control agent also monitors the missive through managing the document agent.
- The document agent extending the messenger agent is charged with the process of missive. It packs and processes the missive over the Internet. The process order of the missive depends on its degree of emergency. If it is a normal missive, the document agent must make a meeting with others prior to make decision about the process order. If it is an emergency missive, the user can deal with the missive directly in time.
- The message agent inherited from the messenger agent provided the communication mechanism. The user of the business department can send messages through the message agent. If any missive arrives, the document control agent will notify the message agent.
- The KBH is the knowledge storage for reasoning mechanism in which the document control agent is committed, describing the services the agent provided, running time application/service, and the specific information.

A user can first use web browser to write a missive, and the agent system can later apply an ECA Parser to parse the business logic of the missive. If the missive passes the parser, the document control agent may get a notification from the agent system and create a document agent to complete the mission of missive. The schedule of missive is determined through a meeting where the document control agent and document agent negotiate with one another. The chairman of the negotiation is the document control agent, and all of the document agents in the same context are the members of meeting. Based on the information regarding waiting time, degree of emergency, and type of missive, the meeting will choose the next document agent to deal with its missive.

4.3. The implementation of the agent system

We use Aglet to implement agent mobility, document life cycle, agent cooperation, agent negotiation, and web-based communication in the document-driven agent system. Aglet [17,20] is designed for development the mobile Java agent system that supports the concepts of autonomous execution and dynamic routing on its itinerary.

- *Agent mobility*: In the agent model, a document agent is a mobile object that has its own thread of control. The document agent is mobile in two different ways: (1) If a user initiate a missive, a document control agent creates a document agent and pushes it from its current host to a remote host; and (2) If a missive can be processed by more than one person at the same time, a document agent will copy itself and move it to other remote hosts.

- *Document life cycle*: The life cycle of a document agent is implemented by the life cycle of Aglet. There are two ways to bring document agent to life: either it is instantiated from creation by document control agent or it is copied from an existing document agent (cloning). To control the population of document agent we must destroy them. To reduce their resource consumption, document agent can go to sleep temporarily, releasing their resources (deactivation) and later can be brought back into running mode (activation).
- *Agent cooperation*: The Aglet server provides an environment for agents to execute in, and the Java virtual machine and Aglet security manager make it safe to receive and for host agents. Multiple agents can exchange information to accomplish the missive transaction. Once a document agent arrives at a local server, it needs to register itself with the document control agent. The document control agent then notify personal agent of its presence.
- *Agent negotiation*: Each document agent is characterized by a priority level relative to other document. If a local server has many incoming document agents simultaneously, then the document control agent must put them into a queue by their priority. The document control agent processes the highest document agent and informs the others to be deactivation. The negotiation mechanism of document control agent is implemented by the multicasting technology of Aglet.
- *Web-based communication*: A user access the Internet through a Web browser. If the user chooses to use the agent system, he simply instructs the browser to connect to a local web server maintained by the system. The user can also use the personal agent to activate the Web browser. The personal can store the user profile. To start a session, a user fills in a special HTTP missive form that is maintained by the Servlet. The Servlet gathers the missive information and passes to the document control agent.

4.4. Discussion

A document agent can pack using the mobile agent technology and document-driven approach, a business document. Thus, the DAB approach provides three properties: (1) trace ability: a function enables an administrator to monitor document processes through the document control agent easily, (2) document life cycle: a feature using mobile agent life cycle to manage document life cycle and concurrent processing, and (3) dynamic scheduling: this function contains two components: a document agent that can dynamically schedule its itinerary, and a document control agent that can dynamically schedule its services.

- *Trace ability*. The agent of DAB is capable of reporting its current state; i.e. total running time, current activity and its status (waiting, deadline) etc. This information is useful for tracing any bottlenecks of the process. For missive sender, it is importance to know where the missive is and what the situations are. The situations of missive contain the arrival time, the finish time, the waiting time, and any exceptional event in a destination department. The document agent routinely analyzes the information in order to plan the next route. We use mobility listener of Aglet implementing the document agent to achieve the trace ability of agent system. The mobility listener can capture the event triggered by mobility reason. The types of event contain dispatching, retraction, activation, and deactivation. We also apply the locker pattern [2], an agent design pattern, to store the private information of the document agent, such as the beginning time, arrival time, and completing time of missive. The document control agent monitors where the missive is and what degree the missive completes through querying the document agent.
- *Document life cycle*. The life cycle of a document is similar to the life cycle of a document agent. Each document agent encapsulates a document and manages its life cycle. And the life cycle of a document agent is implemented by the life cycle of Aglet. It includes create, copy, suspend, resume, and dispose a missive and a document agent. Once an environment gets out of control (e.g. a server of department is crashed), a storm of notification is not sufficient to correct it; more radical steps are necessary. According to the ECA rules of document, the document agent can alter the allocation of task, to transfer a backlog of pending tasks from one user to another, or from one group of users to another. The document control agent can also alter the resource of tasks, to enable more document agents to help clear bottlenecks. A deadline is assigned to each document agent. If the document agent is not completed before the deadline, then it sends an alter message to the document control agent responsible for executing the concerned activity. The document control agent is capable of estimating the total duration time and resources required for the process execution.
- *Dynamic scheduling*. Management of the workflow process is often limited to warnings. It is difficult to dynamically correct a poorly functioning process by, for example, changing priorities or reassigning work [5,28]. Our approach allows dynamic change to procedures by adaptation and negotiation.
  - The document control agent: Every document agent is characterized by a priority level relative to other document agent. This knowledge is used by the document control agent for more efficient task allocation and dynamic scheduling. The document control agent can dynamically change the schedule by perceiving the services, types, and

amount of document agent. So the local server of the department can work more efficiently.

– The document agent: We design the document agent to capture the mobility and the KBH to store the knowledge about the environment. If many missives gathered in a local server of department or a local server of department is restarting, suspended, or shutting down, the document agent may be deactivated or change its regular route to another department and come back to the previous department later. The document agent perceives the environment and responds in a timely fashion to any change that has occurred. Thus, the agent is able to schedule its itinerary and services dynamically. The mobile agent technology is suitable alternative to operate in the Internet that is probably based on the unreliability and low bandwidth net.

5. Conclusion

In this paper, we have proposed an agent system approach, which applied the document-center concept and the mobile agent technology to enable the construction of business process management. The DAB approach can build a document-driven agent system, which encapsulates a single document per agent. In terms of communication, the mobility affects the external architecture of agent system in order to reduce the network loading and to make the communication between agents more flexible. The agent life cycle and the environment sensibility influence the internal architecture of agent system that enables the dynamic schedule.

The DAB provided an architecture and process to develop the document-driven business process management system. It also supplied a mechanism for communication between enterprises or divisions of enterprise. Its strengths include: (1) trace ability: a function that enables administrators to monitor document processes easily, (2) document life cycle: a feature using agent life cycle to manage document life cycle and concurrent processing, and (3) dynamic scheduling: a document agent can dynamically schedule its itinerary, and a document control agent can dynamically schedule its services.

References

[1] A. Abecker, A. Bernardi, H. Maus, M. Sintek, C. Wenzel, Information supply for business process: coupling workflow with document analysis and information retrieval, Knowledge-Based Systems 13 (2000) 271–284.

[2] Y. Aridor, D.B. Lange, Agent design patterns: elements of agent application design, In: Agents' 98 (1998) 108–115.

[3] G. Cabri, L. Leonardi, F. Zambonelli, Mobile-agent coordination models for Internet applications, IEEE Computer Feb (2000) 82–89.

[4] U. Dayal, A.P. Buchmann, D.R. McCarthy, Rules are objects too: a knowledge model for an active object-oriented database management system, in: K.R. Dittrich (Ed.), Advances in Object-Oriented Database Systems, Springer, Berlin, 1988, pp. 129–143.

[5] G. Fakas, B. Karakostas, A workflow management system based on intelligent collaborative objects, Information and Software Technology 41 (1999) 907–915.

[6] T. Finin, Y. Labrou, J. Mayfield, Kqml as an agent communication language, in: J.M. Bradshaw (Ed.), Software Agent, MIT Press, Cambridge, MA, 1997, pp. 291–316.

[7] D. Georgakopolus, H. Schuster, A. Cichocki, D. Baker, Managing process and service fusion in virtual enterprises, Information Systems 24 (6) (1999) 429–456.

[8] D. Georgakopoulos, M. Hornick, An overview of workflow management: from process modeling to workflow automation infrastructure, Distributed and Parallel Databases 3 (1995) 119–153.

[9] A. Geppert, D. Tombros, Event-based distributed workflow execution with eve, In: Proceedings of the IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing (1998) 427–442.

[10] H. Gou, B. Muang, W. Liu, S. Ren, Y. Li, All agent-based approach for workflow management, In: Proceedings of IEEE International Conference on System, Man, and Cybernetics, vol. 1, 2000, pp. 271–292.

[11] K.S. Ho, H.V. Leong, W. Lam, A collaborative word processing system using a corba-based workflow framework, in: G. Blair, D. Schmidt, Z. Tari (Eds.), Proceedings of Third International Symposium on Distributed Objects and Applications (DOA '01), 2001, pp. 176–185.

[12] JATLite. http://java.stanford.edu/java_agent/html/.

[13] N.R. Jennings, T.J. Norman, P. Faratin, Adept: an agent-based approach to business process management, ACM SIGMOD 27 (4) (1998) 32–39.

[14] N.R. Jennings, P. Faratin, M.J. Johnson, T.J. Norman, P. O'Brien, M.E. Wiegand, Agent-based business process management, International Journal of Cooperative Information Systems 5 (2,3) (1996) 105–130.

[15] N.R. Jennings, P. Faratin, T.J. Norman, P. O'Brien, B. Odgers, Autonomous agents for business process management, International Journal of Applied Artificial Intelligence 14 (2) (2000) 145–189.

[16] G. Joeris, Cooperative and integrated workflow and document management for engineering applications, In: Proceedings of Eighth International Workshop on Database and Expert Systems Application (1997) 68–73.

[17] G. Karjoth, D.B. Lange, M. Oshima, A security model for aglets, IEEE Internet Computing 1 (4) (1997) 68–77.

[18] J.Y. Kuo, Mobile agent technology in business processes management, Fu Jen Studies, Science and Engineering 35 (2001) 53–72.

[19] J.Y. Kuo, J. Lee, Xml-based virtual enterprise model: a document-driven agent-based approach, In: Twelfth Workshop on Object-oriented Technology and Application, Tainan, 2001, pp. 103–111.

[20] D.B. Lange, M. Oshima, Programming and Deploying Java Mobile Agents with Aglets, Addison-Wesley, Reading, MA, USA, 1998.

[21] F. Manola, S. Heller, D. Georgakopoulos, M. Hornick, M. Brodle, Distributed object management, International Journal of Intelligent and Cooperative Information Systems 1 (1) (1992).

[22] T.J. Norman, N.R. Jennings, P. Faratin, E.H. Mamdani, Designing and implementing a multi-agent architecture for business process management, In: The ECAI-96 Workshop on Agent Theories, Architectures, and Languages (ATAL-96), 1996, pp. 149–161.

[23] ObjectSpace's Voyager. http://www.recursionsw.com/products/voyager/voyager.asp.

[24] T. Papaioannou, J. Edwards, Mobile agent technology in support of sales order processing in the virtual enterprise, In: The Third IEEE/IFIP International Conference of On Information Technology for Balanced Automation System in Manufacturing (BASYS'98), 1998, pp. 23–32.

[25] T. Papaioannou, J. Edwards, Using mobile agents to improve the alignment between manufacturing and its it support systems, Journal of Robotics and Autonomous Systems 27 (1999) 45–57.

[26] H. Raiffa, The Art and Science of Negotiation, Harvard University Press, Cambridge, USA, 1982.

[27] T. Smith, The future of work flow software, INFORM April (1993).

[28] H. Stark, L. lachal, Ovum evaluates workflow, Ovum (1995).

[29] H. Tarumi, K. Kida, Y. Ishiguro, K. Yoshifu, T. Asahura, Workweb system-multi-workflow management with a multi-agent system, In: Proceedings of the ACM SIGGROUP Conference (1997) 299–308.

[30] W.M.P. van der Aalst, On the automatic generation of workflow processes based on product structures, Computers in Industry 39 (1999) 97–111.

[31] T. Wewers, C. Wargitsch, Four dimensions of interorganizational, document-oriented workflow: a case study of the approval of hazardous-waste disposal, In: The 3lst Hawaii International Conference, vol. 4, 1998, pp. 332–341.

[32] XSL Transformations (XSLT) Version 1.0. http://www.w3.org/TR/xslt.

[33] J. Yen, J. Lee, A task-based methodology for specifying expert systems, IEEE Expert 8 (1) (1993) 8–15.