Document Management in Distributed and Heterogeneous Environments

Dragoslav Pešović[1], Milan Vidaković[2], Zoran Budimac[1], Mirjana Ivanović[1]
[1]Department of Mathematics and Informatics, Faculty of Science, University of Novi Sad
Trg D. Obradovića 4, 21000 Novi Sad, Serbia
[2]Computing and Control Department, Faculty of Technical Sciences, University of Novi Sad, Trg D. Obradovića 6, 21000 Novi Sad, Serbia
dragoslav@im.ns.ac.yu, minja@uns.ns.ac.yu, zjb@im.ns.ac.yu, mirjana.ivanovic@gmail.com

ABSTRACT

Workers, Inc. is a workflow management system fully supported by mobile agents. It is suitable for implementation of different business processes in highly distributed and heterogeneous environments. EXtensible Java-based Agent Framework (XJAF) is a pluggable architecture of the hierarchical intelligent agents system with communication based on KQML. The application of the above-mentioned systems will be considered in the area of Document Management Systems.

Key words: Workflow, Mobile agents, Document management

1. Introduction

Generally speaking, mobile agent is an autonomous program that is able to stop its execution at one node in a computer network, and to transfer itself to another node where its execution continues. Workflow [21] can be defined as the automated part of a business process, organized as a collection of activities, where documents, information or tasks are passed between participants according to a set of procedural rules. A workflow management system (WFMS) provides defining, creating, and managing of workflow instances. The usage of mobile agents in modeling and implementation of a workflow simplifies the workflow management. Workers, Inc. [14] consists of individual agents with autonomous behavior. Mobile agents carrying out workflow instances (so-called workers) have the ability to move to different users, where they can interact with them locally, autonomously taking care of their current position, state, and further itinerary. To allow the exchange of process definitions with various other workflow products, the system had to be made compliant with XML Process Definition Language (XPDL) [22].

EXtensible Java-based Agent Framework (XJAF) [18] is a pluggable architecture and supports pluggable software managers dealing with a particular job. The system is compliant to the FIPA (Foundation for Intelligent Physical Agents) specification and has been implemented using Java Enterprise Edition (JEE) technology.

The application of the above-mentioned systems will be considered in the area of Document Management Systems (DMS).

The rest of the paper is organized as follows. In the next section, the related work is presented. Section 3 describes the architecture of Workers, Inc. Main concepts of agent frameworks and the architecture of XJAF are described in the fourth section. The fifth section briefly discusses an agent-oriented approach to the design and implementation of DMS. Last section concludes paper.

2. Related Work

Several authors have recently suggested a usage of agents in workflow and document management.

In contrary to approaches in [9, 12], Workers, Inc. is highly decentralized and consists solely of individual agents with

autonomous behavior. The only centralized control in our system is the control of user rights to create, access, and change agents and templates. With respect to decentralization, our system resembles [16] that is based on static CORBA (Common Object Requesting Broker Architecture) objects. While decentralization in [16] was one of explicit design goals and had to be explicitly implemented, decentralization in our system comes as a natural consequence of agent mobility.

While in [10] full decentralization of using mobile agents is shortly mentioned, the paper in fact describes the usage of mobile agents in centralized and only for external parties. Our system completely relies on agents and is fully distributed with autonomous agents.

Stromer in [17] describes similar goals and advantages of using mobile agents in as we are, but his implementation is different.

From the problem domain point of view, frameworks can be general-purpose [1, 2, 6], or specialized ones, which solve particular problems [20]. From the technology point of view, agent frameworks are based on either proprietary solutions or on the distributed components technology. Agent frameworks, like JAF (Java Agent Framework) [6], are based on proprietary solutions, while Aglets [1] and JADE [2] are based on the RMI, CORBA and Java EE technology.

The large number of papers is related to the security issues in agent frameworks [3, 8, 20], like: code protection during agent migration, protecting agent frameworks from malicious agents.

This paper presents an implementation of an agent framework in which all important elements are implemented as plug-ins, which provides flexibility in both design and implementation.

Among DMS proposed over the years, there are some that are agent-based [5, 15] or agent-enhanced. However, none of them emphasizes the benefits of agent mobility.

Proposed workflow and document management systems bring some fresh views not only in particular fields of workflow and document management, but in mobile computing as well. But the advantages of decentralized/distributed approach in such systems have not been often recognized.

Our workflow and DMS emphasize the fact that mobile agent has organizational advantages as well. Solutions are easier to program, understand, and maintain, if implemented using mobile agents.

3. Workflow Management System Using Mobile Agents

Workers, Inc. [4, 13, 14] is under development at our University. It is implemented using mobile agents and is especially suited for highly distributed and heterogeneous environments.

Workers, Inc. is envisioned as a community of cooperative agents. The current architecture is two-part, consisting of work-agents (workers) and host-agents (worker hosts). Workers, Inc. is built on top of a Java-based mobile agent system. Agent migration and inter-agent communication benefit from Java RMI and class serialization, and Java sandbox security model is the basis for providing secure agent execution environment. Java API for XML Processing is used for XPDL document parsing.

Process definitions are being completely handled by workers, while the enactment is achieved through the cooperation of a worker carrying a process definition and worker hosts residing at every node of the network. Worker hosts represent central components of the system mediating between the underlying system, workers, and human users.

Workers - Key system component that is encapsulating both the process definition and the execution state of a workflow. While performing a workflow, a worker itinerates among distributed resources carrying process-specific information and autonomously taking care of its execution state. A worker’s behavior is entirely defined by its execution context. When a

worker migrates, its entire execution context is being transported and reconstructed at the target location. The most important part of a context is the worker itinerary (in form of directed graphs), which represents a flow of a worker through a network. To allow concurrent activity execution, agent social abilities are employed. When a single thread of control needs to split into two or more threads, which can be executed in parallel, the worker context is cloned and multiple worker instances are allowed to be executed simultaneously. On the other hand, when multiple parallel threads of execution need to converge into a single thread, agent coordination mechanisms and synchronization techniques are employed.

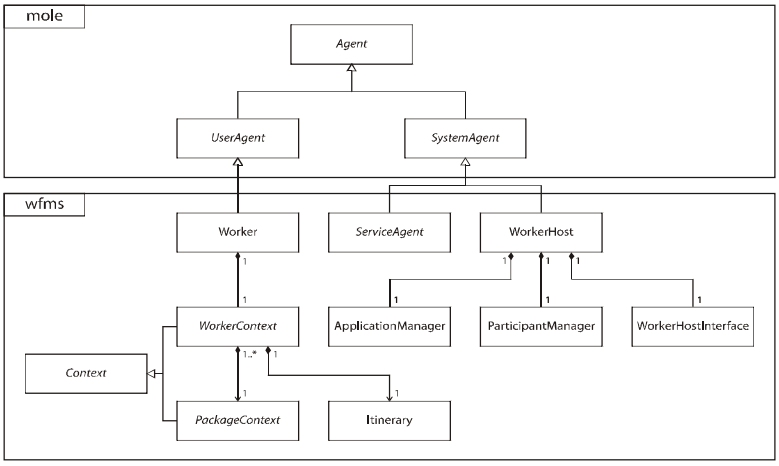

Figure 1. The architecture of Workers, Inc.

Worker Hosts - Every node in the network contains a worker host - a stationary system agent with privileges for the access to host system resources. A worker host spends most of its lifetime receiving requests from workers or users and coordinating their actions. There are three main subcomponents of a worker host: an application manager, a participant manager, and a user interface.

Other Specialized Agents - Workers may need additional services to finish their work. Those services cannot be embedded directly into the workers to keep them as small as possible. Services are implemented, as specialized stationary agents.

Workers, Inc. is a fully distributed system, without central administration, control, and maintenance. All reports, control, and management can be achieved by creating and sending specialized agents that will communicate with other agents in the system and achieve the intended results.

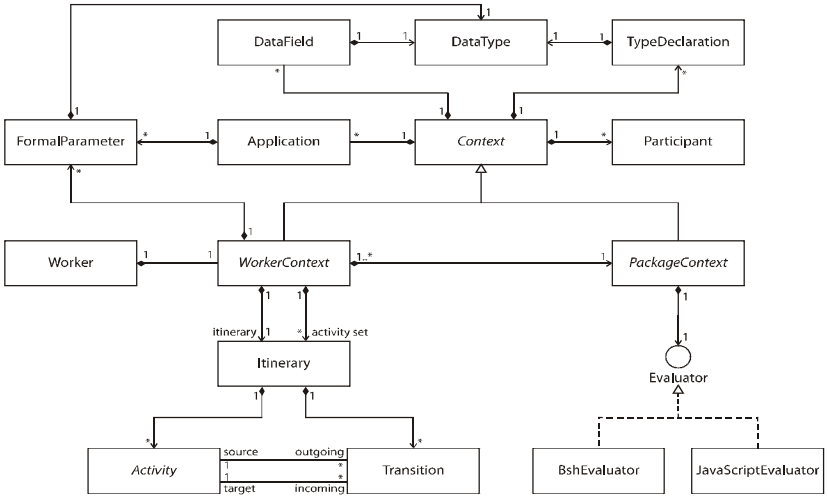

Figure 2. Worker and its contexts

3.1 Worker Execution Contexts

The design of an execution context is done so as to comply with the workflow meta-model specification. From the control-flow perspective, the itinerary is the most important part of a context.

Itinerary - The itinerary has the structure of an arbitrary complex directed graph, where vertices of the graph represent process activities, and edges of the graph correspond to process transitions.

Activities - An activity is the smallest, atomic unit of work in a business process.

Transitions - Transitions connect individual activities. A transition may contain a condition which must be fulfilled for the worker to start performing the target activity. If the transition does not contain a condition, the worker will start the target activity immediately after the source activity has been completed. If the performer assigned to the target activity is different than the one of the source activity, the worker will first transfer itself to the appropriate node in the network, before it starts the activity.

4. Agent Frameworks

Agent technology represents one of the most consistent approaches in distributed systems implementation. Besides solving the problem, agents utilize a certain degree of intelligence and autonomy that are needed to solve the problem. Agent framework represents programming environment that controls agent life cycle and provides all necessary mechanisms for task execution (communication, agent mobility, services and security). An agent framework also provides agent mobility and security (security mechanisms which protect both agents and frameworks.)

Most of the existing agent frameworks are implemented using Java. Such frameworks usually use RMI (Remote Method Invocation), CORBA (Common Object Request Broker Architecture) [11] and Java EE (Java Enterprise Edition) [7] for distribute code execution.

4.1 XJAF

The EXtensible Java-based Agent Framework (XJAF) [18] consists of clients and facilitators. The clients refer to the facilitators for task execution. The task is being executed by the agents engaged by the facilitator. The Figure 3 shows the link between a client and an agent framework.

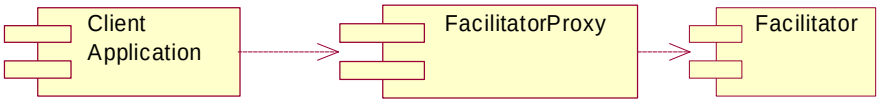

Figure 3. Client and Agent framework link

The client assigns the task to the facilitator; the facilitator engages an agent to execute the task and returns the result to the client. The FacilitatorProxy class ensures that the client application can access the facilitator. It also hides all techniques necessary for work with agents from the client. The client only needs to create an object of the FacilitatorProxy class and to pass it the class representing the task or the KQML (Knowledge Query and Manipulation Language) message, as well as the corresponding listener, which would notify it of the result.

Extensibility of this framework is based on the plug-ins, realized as pluggable managers. The facilitator forwards the parts of its job to the corresponding pluggable managers. The managers are instances of classes implementing the corresponding managerial interfaces. The AgentManager interface is responsible for allocating and releasing agents. The TaskManager interface manages the tasks. The MessageManager interface is responsible for interagent communication. The ConnectionManager interface manages facilitator connection and

relations. The SecurityManager handles security of inter-agent communication. The Figure 4 lists all the managers in the framework.

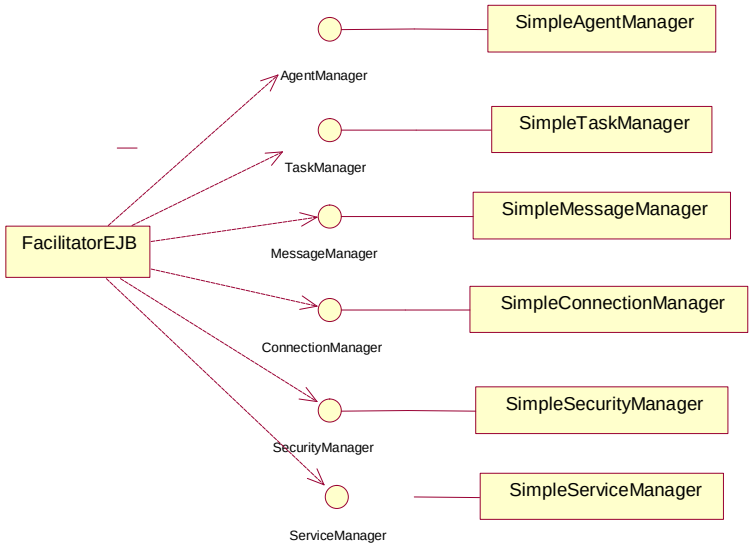

Figure 4. Functionality of individual parts is assigned to managers

Agent Manager - Agent management is done using the AgentManager component. This component is also used as an agent directory. This manager also keeps track of all local agents required by external facilitators, and of all agents that have been moved to another facilitator.

Task Manager - The TaskManager component manages tasks to be performed by the agent framework. It also provides a way of notifying the client about the task execution progress.

When executing a task by sending a KQML message to the agent, the client application sends the KQML message to the Facilitator component. This component looks for the appropriate agent and sends the message to it. When the task is completed, the agent replies to the original message and the message is forwarded to the client using the FacilitatorProxy component.

Message Manager - The agents actually exchange KQML messages. Messages are encapsulated in the base class KQMLMessage. All the communication is done by the MessageManager component.

Connection Manager - The ConnectionManager component defines an inter-facilitator connectivity mechanism. This mechanism defines how separate facilitators form a network.

The facilitators form a certain hierarchy structure. The Figure 5 shows this organization.

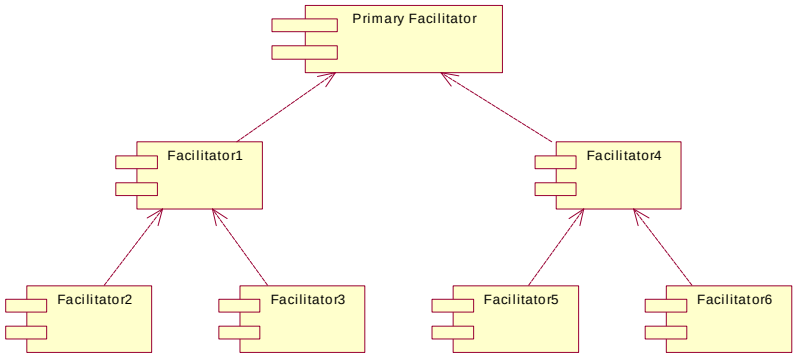

Figure 5. Component diagram of facilitator hierarchy

Security Manager - SecurityManager component [18] handles security issues. It provides encryption, decryption, signature generation and verification for all

messages passing through the framework. Also, this manager handles access to local resources.

Service Manager - The ServiceManager component implements service directory subsystem. It manages the set of services available to agents. The ServiceManager component includes the service repository which holds all available services. Services can be added, removed, searched and used.

5. Document Management

A document management system (DMS) [5, 15] is a computer system used to track and store different forms of electronic documents. Document management controls the life cycle of documents in an organization – how they are created, reviewed, published, consumed, and disposed of or retained. It organizes content in a logical way, and makes it easy to standardize content creation and presentation across an enterprise. DMS promotes knowledge management and information mining and addresses the following issues: Location, Filing, Retrieval, Security, Retention period, Archiving, Distribution, Workflow, Creation, Authentication, Content types.

5.1 Agent-Oriented Approach

The usage of software agents in modeling and implementation of a DMS simplifies the document management because most of its parts are uniformly implemented as (mobile) agents:

- User agents to assist individual users (with incorporated access rights). Every user of the DMS would have a devoted user agent to assist him/her in the authoring and access processes. Those user agents would communicate with other agents in the system directly, or create and send specialized mobile agents in order to achieve the intended results.
- Specialized agents for document retrieval, indexing, archiving, etc. Those agents may be mobile or stationary, depending on the nature of the task they are intended to accomplish.
- Workflow agents to support all kinds of workflows within the system. Workgroups can benefit from agents to coordinate their access efforts.

Since the system consists of many autonomous agents, it is easily changed, extended, and improved. It is often needed just to introduce new agents, without the need to change and even to understand the rest of the system.

6. Concluding Remarks

The idea of implementing a DMS involved two modern, attractive and promising fields in computer science: Software agents and Workflow. The approach to an agent framework implementation using the Java EE technology provides for scalability and reliability. This approach offers agent and service directory services, security, message exchange, and agent mobility.

The main characteristics of suggested workflow system are almost full decentralization and distribution of workflow functions. The proposed organization mimics usual user activities in a real flow of work. Moreover, it relieves them (or any centralized control) from the need to know what to do next with the work-agent. Every user takes care only of work-agents that are currently on its node. Where they came from, why they are here, and where they will go later, is not concern of the user.

References

[1] Aglets Home Page, http://www.trl.ibm.com/aglets/, May 2003.
[2] Bellifemine, F., Poggi, A., Rimassa, G.: "JADE – A FIPA-compliant agent framework", Proc.of Practical Applications of Intelligent Agents (PAAM'99), London, April 1999, pp. 97-108.
[3] Binder, W., Roth, V.: "Secure mobile agent systems using Java: where are we

heading?", Proceedings of the 2002 ACM symposium on Applied computing, 2002, Madrid, Spain, ISBN:1-58113-445-2, pp. 115-119.
[4] Budimac, Z., Ivanović, M., Popović, A.: "Workflow Management System Using Mobile Agents", Proc. of ADBIS '99, LNCS 1691, Springer Verlag, Berlin, (Maribor, Slovenia), pp. 169 - 178, 1999.
[5] Ginsburg, M.: "An Agent Framework for Intranet Document Management", Journal of Autonomous MAS, Vol. 2, No. 3, pp. 271-286, 1999.
[6] Java Agent Framework Home Page, http://mas.cs.umass.edu, May, 2003.
[7] Java Enterprise Edition Homepage, http://java.sun.com/javaee, May, 2003.
[8] Kim Tan, H., Moreau, L.: "Certificates for mobile code security", Proceedings of the 2002 ACM symposium on applied computing, 2002, Madrid, Spain, ISBN:1-58113-445-2, pp. 76-81.
[9] Meng, J., Helal, S., Su, S.: "An ad-hoc workflow system architecture based on mobile agents and rule-based reasoning", Proc. of Int. Conf. on parallel, and distributed computing techniques and applications, 2000.
[10] Merz, M., Liberman, B., Lamersdorf, W.: "Using mobile agents to support inter-organizational workflow management", Int. Jour. on applied artificial intelligence 11(6), pp. 551-572, 1997.
[11] Common Object Request Broker: Architecture and Specification. OMG Specification Revision 2.0, July 1995.
[12] Padalkra, A., Nabar, P., Arora, S., Naik, P.: "SWIFT: Scalable workflow management system using mobile Agents", http://www.iitb.ac.in/~pranav/php/ paper.pdf, 2000.
[13] Pešović, D., Budimac, Z., Ivanović, M.: "Towards a Visual Definition of a Process in a Distributed Environment", 2nd International Symposium on Intelligent Distributed Computing 2008, Catania, Italy, 2008.
[14] Pešović, D.: "A High-Level Language for Defining Business Processes", PhD Thesis, University of Novi Sad, 2007.
[15] Roberto, V., Della Mea, V., Di Gaspero, L., Conti, A.: "MANTHA: Agent-based Management of Hypermedia Documents", Proceedings of 6th IEEE Int. Conf. on Multimedia Computing and Systems, Firenze, IEEE Computer Society, vol II, pp. 814-818, 1999.
[16] Sheth, A., Kochut, K., Miller, J., Worah, D., Das, S., Lin, C., Palaniswami, D., Lynch, J., Shevchenko, I.: "Supporting State-wide Immunization Tracking using Multi-Paradigm Workflow Technology", Proc. of 22nd VLDB Conference (Bombay, India), 1996.
[17] Stormer, H.: "A flexible agent-based workflow system", Workshop on Agent-based approaches to B2B, 2001.
[18] Vidaković, M., Sladić, G., Konjović, Z., "Security Management In J2EE Based Intelligent Agent Framework", Proc. of the 7th IASTED International Conference on Software Engineering and Applications, Marina Del Rey, USA, 2003, pp. 128-133
[19] Vidaković, M., Sladić, G., Zarić, M.: "Metadata Harvesting Using Agent Technology", Proceedings of the 8th IASTED International Conference on Software Engineering and Applications, USA, 2004., pp. 489-493
[20] Wilson, L., Burroughs, D., Sucharitaves, J., Kumar, A.: "An agent-based framework for linking distributed simulations", Proceedings of the 32nd conference on Winter simulation, 2000 , Orlando, Florida, pp. 1713 – 1721.
[21] "Terminology and Glossary", Homepage of Workflow Management Coalition, 1999.
[22] Workflow Management Coalition: "Workflow Process Definition Interface – XML Process Definition Language, Version 1.0", Homepage of Workflow Management Coalition, 2002.