IDEA GROUP PUBLISHING

701 E. Chocolate Avenue, Suite 200, Hershey PA 17033, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.idea-group.com

ITP4865

A Method for Implementing Distributed Learning Environments

Fuhua Oscar Lin and Pete Holt
Computing & Information Systems
Athabasca University, 1 University Drive
Athabasca, AB, T9S 3A3, Canada
Tel: 1-780-675-6175, Tel: 1-780-675-6225
Fax: 1-780-675-6186, Fax: 1-780-675-6186
E-mail: oscarl@athabascau.ca, holt@athabascau.ca

Hong Lin
Department of Computer and Mathematical Sciences
College of Sciences and Technology
University of Houston-Downtown, Houston, TX 77002
Tel.: 1-713-2212781
Fax: 1-713-2218086
Email: linh@dt.uh.edu

ABSTRACT

This paper presents a method for implementing agent-based distributed learning systems to increase the functionality and flexibility of the systems and meet the requirements from the resource-oriented nature of distributed learning environments. We model a distributed learning software system as a multi-agent system using an open collaborative agent system architecture. In order to facilitate specifying conversation patterns and implementing conversation managers, we examine the workflow and tasks in distributed learning environments. The security and privacy management issue in agent-based learning environment is also discussed. An experimental example implemented shows the feasibility of the approach.

1. INTRODUCTION

Over the last few years, universities and colleges have demonstrated substantial progress toward the use of the Web to deliver courses. This approach can be referred to as distributed learning or e-Learning or Web-based education (WBE) with which it is not mandatory that the instructor and the learner be in the same physical location at the same time.

The main advantages of distributed learning over traditional classroom learning are its flexibility in time and location for learning, interactivity in learning elements due to the multimedia capability of the Web, and interactions among instructors, tutors, and learners in both synchronous and asynchronous modes. Furthermore, the potential advantages of distributed learning include the access to distributed learning repositories for course authors, personalized course materials for students, and virtual learning communities for students.

To be more successful in distributed learning, we should implement emerging technologies to develop distributed, adaptive, and flexible software to automate educational activities in distributed learning and develop intelligent tools to reduce the workload of educators and overcome deficiencies in distributed learning.

2. CHALLENGES

A modern distributed learning system should be of the following main features: adaptive curriculum sequencing, problem solving support, adaptive presentation, student model matching. Unfortunately, none of the currently available distributed learning systems delivers these advanced functionalities. There are two main reasons for this. One is the complexity of the systems; another is the cost and difficulty in knowledge modeling and management. Software systems for distributed learning are typically complex because they involve many dynamically interacting educational components, each with their own goals and need for resources, and engage in complex coordination. It is very difficult to develop a system that could meet all requirements for every level of educational hierarchy since no single designer of such a complex system can have full knowledge and control of the system. The systems have to be scaleable and accommodate networking, computing and software facilities that support many thousands of simultaneous users that can concurrently work and communicate with each other and receive adequate quality of service support [1]. As pointed out by Paquette in [2], the design method for such systems should be inspired by software engineering approaches. Most of the currently used distributed learning applications are highly monolithic and seriously lacking in flexibility. They have been built to meet the needs of a specific situation that contains both the domain knowledge and behavior embedded through programming. Such systems would not be easily extended. Any small change in the domain knowledge would require an intensive system-wide modification to alter the information and all objects that initiate actions based on that changing information.

3. AGENT-ORIENTED MODELING

Software engineering challenges in developing large scale distributed learning environment can be overcome by an agent-based approach [3]. We can model a distributed learning system as a set of autonomous, cooperating agents that communicate intelligently with one another, automate or semi-automate educational processes, inquire (interact with) human users at the right time with the right information.

A software agent is a software package that carries out tasks for others, autonomously without direct intervention by its master once the tasks have been delegated. The researchers at Intelligent Software Agents Lab at Carnegie Mellon University's Robotics Institute define an agent as: " … an autonomous, (preferably) intelligent, collaborative, adaptive computational entity. Here, intelligence is the ability to infer and execute actions, and seek and incorporate relevant information, given certain goals". What is important to our application is the ability to meet the requirements of distributed learning systems, including (1) Autonomy — the ability of the agent to act on its own volition, but with respect to the needs of the overall software system. (2) Communication — the agent must be able to interact with not only other agents, but also other non-agent sub-systems in order to meet the requirements of the overall system; (3) Knowledge — the agent must be able to gather knowledge and adapt subsequent behavior with respect to its knowledge base. It should 'learn' from its experiences.

Why do we adopt agent-based approach? Why would we think of using them instead of simply writing another subroutine? It probably boils down to three things. First, distributing tasks to numerous specialized, fine-grained modules promotes modularity, flexibility, and incrementality. It lets new services come and go without disturbing the overall system. Limiting the complexity of an individual agent simplifies control, promotes reusability, and provides a framework for tackling interoperability. Second, their autonomous nature makes them a fire-and-forget approach. You don't need to remember to invoke them explicitly at the right point; they are able to react for themselves if they have access to the right data. The central feature of software agents is the ability to independently carry out tasks delegated to them by people or other software. Robust marketplace-like educational resources available today or tomorrow simply could not function without being able to

delegate to software the multitude of tasks that would otherwise be left to armies of people to handle. Third, they are more robust than conventional software. If conditions are less than ideal, they should be able to degrade in a graceful way rather than just failing and perhaps brings down another process.

Agent-oriented software engineering has become one of the most active areas in the field of software engineering. The agent concept provides a focal point for accountability and responsibility for coping with the complexity of software systems both during design and execution [4]. The agent-based approach to developing complex distributed systems has been successfully applied and documented in many domains, including air traffic control, manufacturing, information retrieval, network management, and entertainment [5].

4. AGENT SYSTEM ARCHITECTURE

To meet the requirements from the resource-oriented nature of distributed learning systems, we adopted an open, flexible, and collaborative agent system architecture (CASA) [6-7] to model the agent systems.

In CASA, agents are seen as software entities that pursue their objectives while taking into account the resources and skills available to them, and based on their representations of their environment and on the communications they receive. The collaborative architecture separates the modeling of multi-agent systems from the specifications that designers need to commit given the low-level mechanism of proprietary frameworks used in the implementation of multi-agent systems. The three elementary components identified as fundamental in the design of collaborative multi-agent systems are computer resources, agents, and owners. All agents in the architecture can be categorized into five groups: user agents with control interface; task (application) agents; collaboration agents; knowledge management agents; and resource agents.

4.1 Agent Communication

It is now widely recognized that interaction among agents in a system is probably the most important single characteristic of complex software and agent communication can be better modeled and more easily implemented when a conversation rather than an isolated message is taken as the primary unit of analysis [8]. A conversation is a sequence of messages involving two or more agents and centering on a set of pre-designated topics, intended to achieve a particular purpose. To correctly and efficiently engineer software architecture, we can model and specify agent interactions in distributed learning as conversation schemata, in which interaction patterns and task constraints in collaborative agent systems are constructed within class hierarchies of conversation schemata [7]. These patterns are executed through identifying agent types and investigating interactions among agents at different levels of systems, as well as considering the relevant topics and information to be exchanged. The schemata can be detailed in terms of communicative acts. The dynamic selection, instantiation and execution processes of conversations are then enabled by incorporating conversation managers (CM). One of the advantages of incorporating conversation managers into multi-agent systems is enabling communication-monitoring facility in multi-agent systems [9].

4.2 Workflow, Tasks, and Agents

In order to facilitate specifying conversation patterns and implementing conversation managers, we examine the workflow model of the educational tasks in distributed learning environments.

The lifecycle of a distributed learning course starts from course planning. The next is course development in which the course is designed and developed by a team of course developers consisting of professors, editors, and visual designers. Then the course package is delivered to students via the Web and/or seminars under the coordination and facilitation of an instructor. One or more tutors may be needed for tutoring the course depending on the size of the class. The course continually goes through course evaluation and course revision until it is closed.

From the workflow model of the course development, we can build a collaborative system model that partitions the problem into one or more smaller tasks. First, we have user interface agents for different users in the system: program planner agents, course author agents, instructor agents, student agents, and tutor agents. Second, according to task decomposition, we can have application agents such as: program planning agents, course generation agents, course maintenance agents, course recommendation agents, etc. _Third, for collaboration and communication, we need a set of collaboration agents like Local Area Coordinators and Conversation Managers. Next, we need knowledge management agents to manage domain knowledge (ontologies, concepts, etc.), knowledge about students, knowledge on tutoring, and knowledge about environment. Finally, we have resource agents that include course developer information agents, learning object directory agents, instructor information agents, tutor information agents, and student information agents. These information agents are responsible for getting information about resource needed.

4.3 Security and Privacy Management

Collaborative agent system architecture is a fully distributed multi-agent system built on broadband-based Internet. Therefore, to manage overall security and protect all communications among agents are necessary. First of all, security issues cover all security problems related to network (esp. Internet) technology. These security problems contain denial of service for e-learning systems, gathering information from the data delivery, and unauthorized access to the private resource or information in the e-learning systems. These problems have been well protected using the existing security technologies such as PKI for authentication, secure-IP data delivery, and intrusion detection for unauthorized access or denial of service. The main issues in collaborative agent system architecture are privacy and security management for Web-based applications.

Second, privacy is defined as the interest that individuals have in sustaining a 'personal space', free from interference by other people and organizations. Information Privacy in collaborative agent system architecture is the interest an education institute (agent) has in controlling, or at least significantly influencing, the handling of data about themselves. To protect privacy is to find appropriate balances between privacy and multiple competing interests. Recently, several techniques have been developed for providing privacy protection. Generally, they can be divided into two kinds of techniques: one is for providing pseudonym protection; another is for providing anonymity and unobservability using anonymous communication networks.

In our study we are using a policy-based privacy and security management scheme for managing security and privacy for e-learning systems [17]. The policy-based security and privacy management can be realized in the collaboration agents. The collaboration agents will provide a policy management system for the user to create, upgrade, and manage the policies using policy specification language.

5. AN EXAMPLE OF IMPLEMENTATION: AGENTS FOR COURSE MAINTENANCE & RECOMMENDATION

Online course materials for distributed learning typically include all of textbook or e-book and study guide, plus the project description. They can be downloaded and installed on students' hard drive. That means students only need to be online when doing the Internet explorations, posting material on the conference or doing the quizzes, plus of course sending or receiving e-mail. The online course materials are updated often in order to keep them as current as possible, esp. in some rapidly changing fields like 'computing and information systems'. Because of the complexity of the materials, and the short development cycles within which they are produced, our best efforts are sometimes not adequate to prevent occasional errors from slipping through, and students should therefore be prepared to encounter the odd minor 'glitch' in online courses. Therefore the course instructor should make the necessary adjustments for the benefit of students. Whenever there's a significant change the content of several designated web pages of online

course materials, students who are interested in the topic and all students taking the course will be notified by the course coordinator by e-mail.

From the interaction analysis, we get a conversation model of the course material change notification, which consists of the following elements:

Notification Agent Control Client: The Notification Agent Control Client of an instructor or a student runs on his/her machine and allows him/her to control the behavior of the corresponding Notification Agent deployed in a distributed environment. For example, a message editor is designed to allow the instructor to define the format of the messages to the students by the Notification Agent.

Notification Agent: The basic function of the Notification Agent is to send e-mails to students taking the course according to the student profiles stored in a database when the course material has been significantly changed. The Notification Agent is able to contact the distributed learning database on the server side.

Conversation Manager: A conversation manager is designed for monitoring and coordinating all activities in the schema to ensure the quality of service of the agent system. The conversation manager is usually running with a virtual Yellow Page.

Knowledge Management Agent and Topic Tree: The essence of agents is taking task knowledge now contained in the heads of the workforce and codifying it into software systems. We are building a package to implement a rule base reasoning mechanism and the only reasoning mechanism implemented is forward-chaining. We are developing the agents in a way that the domain knowledge (e.g. topic tree) and the behavior components (knowledge engine) of the agents are physically separated so that either can be changed without impacting the other.

Monitoring Agent: The Web Change Monitoring Agent of a system administrator monitors a collection of course material URLs stored in a database. It scans the given pages periodically. When the agent detects a significant change, it sends a message to the Notification Agent.

Student Information Agent: A Student Information Agent is designed for providing services about student information, such as providing an e-mail list for a course by automatically maintaining the email list of students taking a course.

Student Information Database: The database resource includes a student information database, an instructor information database, and a link database.

Ontologies are fundamental to domain knowledge modeling of the agents for WBE to index, organize, and communicate about both the administrative and instructional design elements including programs, courses, learning objects, questions, test questions, and discussion messages and the things associated with them.

A 'topic tree' for the 'computing and information systems' courses is part of domain knowledge. For example, it is used for a student to represent his/her interests in the course materials so that his/her agent can notify the student of significant changes if any.

We have implemented in Java the agents for the link maintenance and recommendation by using the modeling approach proposed and the CASA architecture.

6. SOME RELATED WORK

In [10], Greer, *et al*., (2001) elaborated the lessons learned from several large-scale real world deployments of the I-Help [11] agent-based peer-help learning support system. The software engineering lessons learned by them are useful for us to deploy a complex system in real world for a large number of users. Gavrilova et al. described in [12] a project of an intelligent multi-agent system for distance learning using Learner-model approach. Paper [13] conceptualized three types of intelligent agent to assist teachers and learners. Thaiupathump *et al*.' (1999) investigated the effects of applying intelligent agent techniques to an online learning environment [14]. They created the knowbots that automated the repetitive tasks of human facilitators in a series of online workshops. The findings indicated that the use of knowbots was positively associated with higher learner completion rates in the workshops. Paper [15] identifies the roles of agents in distributed educational activities. Baylor defined three major educational potentials for agents as cognitive tools: (1) As assistants, managing information overload; (2) serving as a pedagogical expert; and (3) creating programming environment for the learner [16].

7. CONCLUSIONS & FUTURE WORK

We have presented a method for developing agent-based systems for distributed learning to increase the functionality and flexibility of the systems and meet the requirements from the resource-oriented nature of distributed learning environments. We model a distributed learning software system as a multi-agent system using an open collaborative agent system architecture. In order to facilitate specifying conversation patterns and implementing conversation managers, we examine the workflow and tasks in distributed learning environments.

In the future, we will be working on knowledge modeling and management for constructing agents for adaptive distributed learning. We will develop models and techniques of knowledge modeling and management, focusing on the specification of static and dynamic knowledge resources in educational information systems.

Security and privacy issues are critically important to agent-oriented systems for distributed learning. We have been studying these issues. We will implement the policy-based security and privacy management for the CASA-based e-learning systems. The performance about the quality of service of the agents will be also studied by using a simulation approach.

REFERENCES

[1] Vouk, Mladen A., Donald L. Bitzer and Richard L. Klevans, Workflow and End-User Quality of Service Issues in Web-Based Education, IEEE Trans. on Knowledge and Data Engineering, 11, (4); July/ August 1999, pp. 673-687

[2] Paqutte, G., Implementing a Virtual Learning Center in an Organization, ITHET-2001, Kumamoto, Japan, July 2001

[3] Vassileva J., Deters R., Greer J., MaCalla G., Kumar V., Mudgal C., (1998) A Multi-Agent Architecture for Peer-Help in a University Course, Proceedings of the Workshop on Pedagogical Agents at ITS'98, San Antonio, Texas, 64-68

[4] Yu, Eric, Agent-Oriented Modelling: Software Versus the World, Agent-Oriented Software Engineering AOSE-2001 Workshop Proceedings. LNCS 2222. Springer Verlag. 206-225.

[5] Wooldridge, M. and N. Jennings, Intelligent Agents: Theory and Practice, The Knowledge Engineering Review, 10(2), 115-152, 1995

[6] Flores, R.A., Kremer, R.C., & Norrie, D.H. An Architecture for Modeling Internet-based Collaborative Agent Systems, in T. Wagner & O.F. Rana (Eds.), Infrastructure for Agents, Multi-Agent Systems, and Scalable Multi-Agent Systems, LNCS1887, Springer-Verlag, 2001, 56-63.

[7] Lin F., Norrie D. H., Flores, R.A., & Kremer R.C. Incorporating Conversation Managers into Multi-agent Systems, in M. Greaves, F. Dignum, J. Bradshaw & B. Chaib-draa (Eds.), Proc. of the Workshop on Agent Communication and Languages, 4th Inter. Conf. on Autonomous Agents (Agents 2000), Barcelona, Spain, June, 3-7, 2000, pp. 1-9.

[8] Bradshaw, J. M., Software Agents, Menlo Park, CA: AAAI/ MIT Press, 1997

[9] Lin, F. and Korba, L., (2000). "Incorporating communication monitoring facility in multi-agent systems", Proc. IEEE Int. Conf. on Systems, Man, and Cybernetics (SMC'2000), 8-11 October 2000, Nashville, TN, 3116-3121.

[10] Greer J., McCalla G., Vassileva J., Deters R., Bull S., Kettel L. (2001) Lessons Learned in Deploying a Multi-Agent Learning Support System: The I-Help Experience, Proceedings of AIED'2001, San Antonio, 410-421.

[11] Greer, J., McCalla, G., Cooke, J., Collins, J., Kumar, V., Bishop, A., and Vassileva, J.: The intelligent helpdesk: Supporting peer-help in a university course. In: Goettl, B. P., Halff, H. M., Redfield, C. L. and Shute, V. J. (eds.) Intelligent Tutoring Systems. LNCS1452. Springer-Verlag, (1998) 494-503

[12] Tatiana Gavrilova, Alexander V. Voinov, Irina Lescheva:

Learner-Model Approach to Multi-agent Intelligent Distance Learning System for Program Testing. IEA/AIE 1999: 440-449

[13] Jafari, A., "Conceptualizing Intelligent Agents For Teaching and Learning," International Conference on Intelligent Agents, Las Vegas, USA, July 2001.

[14] C. Thaiupathump, J. Bourne, J. O. Campbell, Intelligent Agents for Online Learning, JALN 3(2) - November 1999

[15] Lin, F., Holt, P., (2001). "Towards Agent-based Online Learning", IASTED Int. Conf. Computer and Advanced Technology in Education (CATE), June 27-29, Banff, Canada, pp.124-129.

[16] Baylor, A. (1999). Intelligent agents as cognitive tools for education. Educational Technology, Volume XXXIX (2), 36-41.

[17] Yang, C. Lin, F., and Lin H., "Policy-based Privacy and Security Management for Collaborative E-education Systems", *Proceedings of CATE 2002*, Cancun, Mexico, pp.501-505