A Learner-Centered Distributed Learning Environment

Peter Holt, Oscar Lin, Harris Wang
CCIS, Athabasca University, 1 University Drive, Athabasca AB T9S 3A3, Canada

Qiangguo Pu, Nikos E. Mastorakis*
Computer Center, University of Science and Technology of Suzhou
298 Binhe Road, Suzhou, Jiangsu 215011, China
*Hellenic Naval Academy and WSEAS, Piraeus, 18539, Greece

Abstract: - One of the factors contributing to the rapid growth of e-Learning has been the rapid development of the World Wide Web relative ubiquity of standard Web-browsers. However, we argue in this paper that the standard Web-browser is reaching its limits as the de facto client for e-Learning. The standard Web-browser-central server combination in many circumstances does not deal well with factors such a performance, security, privacy, and learner-centered control. We outline an alternative framework to support the evolution of agent-based multimedia e-Learning.

Key-Words: - E-Learning, learner environment, privacy, agents, distributed

1 The Distributed Learner Platform/Module
One of the attractive features of the web is the fact that information can be accessed with a single tool familiar to the student (the Web browser). Web-based instruction has primarily been based on server-side solutions (e.g. Web-CT, Blackboard) supporting a standard browser. However, we propose a flexible modular architecture that distributes functions across the Web and assumes that much processing will occur on the powerful student computer external to the browser [7]. Next we provide a rationale such a distributed approach.

2 Rationale for a Distributed Platform
2.1 Performance
For a moment let us suppose perform a thought experiment. Imagine there are 500 computer science students simultaneously compiling Java programs. Imagine that these students can compile them on their own computers or can compile them on a central server. From a performance viewpoint, what is most likely to be best both for the students compiling the programs and for the other students?

As another example, there has been considerable exploration of agent technology applications for education: a multi-agent approach to the design of peer-help environment (Vasileva et al, 1999); agents for information retrieval [11]; agents for student information processing, distribution, and feedback collection [14]; pedagogical agents [16]; tutoring agents [21]; agents for assignment checking [19]; agents for student group online support [26]. As a conservative estimate imagine in the future there will be 10 intelligent agents specific to student functions. Presumably most of these agents could run on a central or be distributed to the students’ computers. Again let us perform a thought experiment with 10000 students simultaneously running their ten student agents on central servers or upon their own platform. From a performance viewpoint what is most likely to be best both for the students running their agents and other students? Greer (2001, personal communication) reported that much of the “intelligence” in I-help [23] had to be turned off for performance reasons.
We could select many other examples but the point is that most E-Learning systems currently assume too much of the processing will occur on the central server and learners will require only a thin client. This assumption cripples people’s vision of E-Learning by both constraining the types of applications that can run on the client side (browser dependent) and server side (performance dependent).

2.2 Human Computer Interaction

2.2.1 Privacy/Security

Institutions collect a variety of information about the student. Some of this information they must protect from other students. Other information students may wish to share to facilitate the learning process. Students are anxious about privacy and security of their data [8]. If the student perceives her information to be secure and private she be less anxious. Also, it appears the more familiar a person is with a system the less likely it will be to arouse anxiety [12]. Furthermore, people like to control their own agents and learning parameters. The less personal material that is shipped over the Internet the easier it is to ensure privacy [17]. Thus it appears that maximizing the amount of information and supporting technology such agents upon the students local platform will minimize anxiety. This is important because complex learning occurs better when the learner is not anxious.

2.2.2 Control

Computer users suffer from a perceived loss of control, alienation, inadequacy, loss of responsibility and damage to self-image [1]. People like absolute boundaries, but they like to be able to make choices on the own within those boundaries. The learner should be able to over-ride most instructional choices made for the learner. For mature life-long learners the boundaries should be absolute constraints of the system, not boundaries imposed by the performance or knowledge of the learner. A common comment [12] is that learner’s should be able to turn agents off or on at their discretion. Users learn quickly and gain a fast sense of mastery when they are placed ‘in charge’[22]. The mature lifelong learner will be curious about their learning environment. The learner should be able to inspect the system. It is easier to provide a learning environment within which the learner feels safe and secure, if that learning environment is embedded within the learner’s own computer.

2.3 Complex Interactive Multimedia Learning Objects and Tools

2.3.1 Multimedia and Simulations

Researchers are beginning to emphasize that e-Learning must include more than what is currently encompassed by browser-based learning. In designing electronic materials, it often seems course authors cannot get beyond the simple textbook paradigm. Although electronic texts do have some advantages in terms of updating, searching, indexing, and integrating with other learning objects, this format does have its limitations. However, when people become frustrated with the limitations of online texts they frequently only want to supplement it with canned video of a lecture. Unfortunately, even interactive video can replicate a poor learning experience in a new medium if not properly integrated into the learning experience.

Pre-web computer based training and intelligent tutoring systems have explored simulations for decades. In computer science at least two discrete-event modeling and simulation packages, similar to the Simula-67 package DEMOS [2] are implemented in Java [13]; Little, 1999: http://javasim.nel.ac.uk). The CAREO project (www.cancore.ca) defines learning objects to include "simulations, tutorials, drill and practice modules, content databases, multi-media exercises" [6]. While current browser-based learning has advantages in availability and standardization, those advantages are becoming less important with the growing literacy of the population and the standardization of human computer interfaces. Bandwidth limits how far one can
go with a standard server – generalized browser client model as the main sandbox for E-Learning for more complex learning materials. Also there are other reasons that applications such as 3-D gaming do not run within a generalized browser. Working within the browser sandbox incorporating integrated multimedia will be ineffective and due to the problems of coordinating the components of multimedia. The plug-ins for audio, video, and 3D animation are not easy to coordinate with browser text. Multimedia solutions with the browser such as macromedia Flash will be too limited for comprehensive effective e-Learning. Ultimately what will be required for e-Learning is a framework on the learner’s platform (or with his wall or within her desk) that can coordinate and talk to multimedia learning objects and players. The learning objects themselves will have to provide information about the relevant parameters for their coordination. Some of this may be done at the time a course is designed and packaged. Other coordination may be controlled at the time of delivery of the course by the learner or by the learner’s agents.

2.3.2 Integration of Discipline Specific Tools
It is very useful for computer science students to use a variety of tools on their computer. A thin client might work best for remedial drill in a junior fact-based course. However, peer-to-peer technology [20] may be better for teaching some applications of distributed. Similarly, learning programming is better served by integrating instruction with processes running on the student platform.

2.3.3 Learner Support Tools
An integrated course delivery system should provide necessary tools for students to manage their learning. These include tools to manage their working files, keep track their study schedules, or even help students to lan. This becomes very useful especially for students who are full time employed and have only fractions of time for their course work. Students are used to working on their own platform with these types of tools and in fact the tools might function for more than one institution in an International learner tool package.

3 Design of Whole System
3.1 Integrated Architecture
It is important that there be an overall integrating architecture for organizing the various functions and the associated technology. One version of an integrated system for Web-based distance education is based on a set of modules [25]. The modules are a high level description of the architecture of the system and much of the functionality within each module will be implemented with the agent technology [18]. The modules proposed include server side modules for managing enrollments, managing collaboration, distributing course materials, and other functions best done for an individual institution on a central server. Other modules may run on the tutors’, learners’, or course developers’ platforms. The modules each have their own set of agents and may include other software tools. Conceptually agents and tools within a module share data more easily than agents in different modules. The modules may include other tools. In this paper we focus on the Student Learning Module as it best exemplifies the distributed approach.

3.2 Student Learning Module (SLM)
This module located on the learner’s computer is designed to help students digest course materials, access external documents, collaborate with their fellow students, contact their tutors for helps, and submit their assignments. It includes a multimedia viewer and embedded browser and tools for managing learning. It provides students with a built-in interface, as a client of the course collaboration module, to collaborate with their fellow students, process multimedia materials, and view XML/HTML pages. In addition, SLM provides features that allow mobile users to learn on the move. We are beginning the
definition of this interface. The architecture is supported by an agent class archive, student profiles (for student modeling) and access to course materials. An institution’s student agents reside primarily on the student client after being dispatched from an agent archive on the server. We anticipate an agent corps that will perform a variety of tasks for the learner. Some agents may be mobile [24] and visit a variety of hosting institutions and perhaps other learners. Some agents may belong permanently to the learner and not be associated with any particular institution.

Agents on the learner platform from a particular institution can be described as Participant agents or Assistant agents, or Background agents according to their roles and functions. These agents may be from a particular institution or may be generic.

Participant agents can be further classified into Teaching agents and Learning agents. Both have roles in learning activities. Teaching agents are also called pedagogical agents. Two typical applications of pedagogical agents are Intelligent Tutoring Systems and Learning Companion Systems [4]. Intelligent Tutoring Systems employ one-to-one tutoring activities, thus they are authoritative simulated teachers who are domain experts, tutoring, coaching, or guiding the learner [15]. The simulated teacher can provide adaptive instruction appropriate to the individual learner according to the learning process of the learner. They may generate tests of random items, tailor the course material to the student model and manage course material presentation.

“Learning companion”, “co-learner”, “simulated student”, “virtual participant”, and “artificial student” are names used by different researchers to denote a non-authoritative teaching agent, simulated by a computer. To the student, a learning companion is not a domain expert and may make mistakes. It can act as a peer tutor [5] tutee or teachable student [3], collaborator [10]. Learner’s assistant agents serve as personal assistants, cognitive tools, intelligent tools of the learning material, or surrogates on behalf of the learners in their absence. As an autonomous entity, an assistant agent may share a degree of control of his or her own behavior from its owner. A learner assistant agent provides each learner with individual and useful information that pertains to learning activities such as searching for learning objects or classmates. For example, in the PhelpS system, a student assistant agent helps a user record mistakes that have been completed and to obtain help from peers when required [9].

Background agents include agents responsible for maintaining a student model, negotiating with foreign agents on the server side, negotiating with guest agents of local learning objects, keeping the student’s course learning materials current, and keeping the supporting agents current [18]. The latter functions require negotiating with server-side agents. If the student model is universal, agents maintaining a student model might not be associated with a particular institution.

4 Implementation
4.1 Issues and Strategies
Neither a single department nor a single university will be able to support a complete open learning environment. We are involved with or track a number of open source systems developed by formal and informal collaborations of educational institutions and government. This allows us to keep our options open and it avoids the control of our e-Learning environments by commercial interests. JA-SIG (http://www.ja-sig.org/) uPortal is a free, open standard, Java/XML based sharable portal for post-secondary institutions. The Open Knowledge Initiative (http://web.mit.edu/oki/) is developing open source software for e-Learning. The Mozilla project (http://www.mozilla.org/) is
developing an xml-enabled browser that can be embedded in java. Project JXTA (http://www.jxta.org/) is an open source effort aimed at providing “an open, generalized protocol for distributed applications. GNOME (http://www.gnome.org/) is an open source desktop environment for Linux.

The agent corps will be critical for automating updates of course materials and versions of the software. For agents the content of the communication message is represented as a restricted form of XML. All agents are required to include an XML parser. An XML Schema is defined to restrict the syntax of commands and their parameters. The schema is considered part of the agent package. Each agent keeps a list of other agents to which it will talk. Often the agent cannot communicate “directly” to other agents or users. For example if an agent runs on a secured server, the server would not allow a direct socket between that agent and an agent on the student platform. Instead “Command Forwarder” and “Message Dispatcher” interfaces are used. Implementation of both Command Forwarder and Message Dispatcher are platform-dependent. All inter-machine requests from an agent to another go through a Command Forwarder. The recipient agent interprets the requests and “decides” whether to perform them. Control interface controls the agent through a “command”, which is simply a text String with a few text parameters created on the control interface. To interpret the command each agent keeps a Command Set. Default command set is available but specific implementation is possible by overriding the default command set. With different interpretation of the same command String it will be possible to design different behavior of agents. Agents send “messages” to the control interface and all observers if its status is changed or if any activities are running. Though it is not normal, it is also up to the recipient to interpret the message.

Currently, we have a working prototype of the basic framework but we have not implemented the complete agent corps on the student platform nor tested course delivery in a live course.

References:


