MULTI-AGENT DESIGN IN FLEXIBLE LEARNING ENVIRONMENTS

Tom Fenton-Kerr, Steve Clark, Graham Cheney, Tony Koppi and Marcel Chaloupka

Centre for New Technologies in Teaching & Learning (NeTTL),
University of Sydney, Australia
email: tfk@nettl.usyd.edu.au
http://nettl.usyd.edu.au

ABSTRACT

Educational software programs have become increasingly complex in both scope and concept. Learners need assistance in navigating and understanding the information they present. This situation requires more than simple text-based help pages, hence the development of pedagogical agents, programs that make use of artificial intelligence approaches to provide timely, contextual help or instruction to a learner. A pedagogical agent may adopt a number of different roles in fulfilling its task, taking the role of a guide, a prompt, or provider of definitions or explanations of a procedure. Additionally, such an agent may need to adopt different modes of representation to provide the most effective form of communication. Tutor designs that attempt to encompass many disparate programming aspects such as the above can suffer from a level of complexity that is difficult to manage effectively. In trying to be all things to all users, they can have a negative effect on the learning process. This paper suggests a solution: a modular approach to the design of pedagogical agents in general. A multi-agent learning environment is discussed, together with the role that such a scheme has in mediating interactions between teachers, learners, their peers and the courseware used in a collaborative learning environment. Discussion will concentrate on the role of multi-agent design in a learning environment, effective modes of agent representation, and the contribution of such an approach to flexible learning.

KEY WORDS

Intelligent tutors, pedagogy, agents, learning systems, design.

1. INTRODUCTION

Learners accessing educational software programs can be overwhelmed by the amount and complexity of the content, which can have a negative impact on the learning process. Increasingly, they need assistance in navigating and understanding the information such programs present. *Pedagogical agents* are program modules that make use of artificial intelligence approaches to provide timely, contextual help or instruction to a learner. Intelligent tutor schemes may employ multiple agent modules, each given a unique task that satisfies part of an overall learning objective. A tutor system that incorporates a multi-agent design can give added flexibility to both users and program designers. The agents that make up the tutorial part of such a learning program are integral and, in fact, essential to the learning process. They are motivated to fulfil the program's learning objectives by motivating a learner appropriately. Rich & Sidner (1997) call this collaborative approach, where agent and learner share goals, a *shared plan*. This paper explores such approaches by describing a proposed architecture for implementing a multi-agent help system within the context of a *distributed collaborative learning environment,* described in the next section. The contribution of such an approach to flexible learning in general is then discussed, concluding with remarks about future uses and representations of pedagogical agents in learning environments.

2. MAIN DESIGN ISSUES

2.1 THE LEARNING ENVIRONMENT

In order to successfully design and implement a multi-agent help system within a computer-based learning environment, a number of key issues need to be addressed at the outset. Core questions such as "What are the learning objectives for this course? What is the range and type of material to be covered?" can give an instructional designer some idea of how to design a help system that will aid a learner to accomplish the learning task. Another obvious question would probably be "What specifically is the learning environment that will require help from the proposed system?" In the case of our own institution's projects, this is usually a distributed collaborative learning environment, which offers users many opportunities for cooperative learning and exchange across a network. The elements of this environment, some of which are likely to have a direct bearing on the design of a multi-agent help system, may include any or all of the following:

- Sequential pages of course content, linked by either navigation objects or hypertext.
- Web-based multimedia components (Animations, graphics, video, audio and text).
- Asynchronous peer-to-peer and peer-to-instructor communication links (email, threaded discussion groups).
- Synchronous communication links (chat rooms, shared electronic whiteboards, audio and/or video conferencing).
- Interactive learning elements involving animations, dialogues, forms, speech recognition and text-to-speech communication.

As can be easily seen from the above list, an on-line learning environment can quickly become a complex terrain, somewhat daunting and overwhelming to the novice learner. Offering a user many choices about where to go and what to do carries its own penalties in terms of navigation and learning issues such as "knowing (and keeping track of) what you know". This is where a multi-agent help scheme can make a difference to the learning experience; by helping a user to navigate the material effectively, keeping track of where they are, and prompting the user into effective action. This job-description might encourage us to ask what range of agent abilities and actions are required of the proposed multi-agent system, described in the section below.

2.2 AGENT REQUIREMENTS

Conceptually, pedagogical agent designs cover a large range of approaches, each requiring the agent or agents to implement a variety of skills in accomplishing the task of providing effective help to a learner. When deciding on the scope of agent abilities we looked first at the course objectives and the proposed learning environment, which gave us some ideas on the types of skills our agents needed to have. Some on-line courses are sufficiently simple and direct in their implementation to require only a basic help system. Such courses utilize a single graphical interface agent, with local knowledge about the current page, and the ability to give simple spoken descriptions of any page elements. Their usefulness is limited, and consequently they tend to have limited abilities beyond scripted responses to a user query.

More ambitious courses, which offer access to large databases of resources and multiple modes of interaction, such as assessable courseware and communication facilities require help systems that go beyond simple descriptions. An agent may need to demonstrate the use of a particular media-type such as a video control, using a learning-by-demonstration approach. In a large course consisting of multiple hyper-linked pages the ability to offer contextual help becomes more critical. To do this effectively a pedagogical agent needs to track the user's passage through the course, and to know what the content of the current page is. If the agent's task is to offer remedial help it needs to make inferences about the learner's current level of understanding, sometimes by inferring a learner's knowledge-state from tracking results, or by referring to a log of the learner's previous interactions with the software. The various activities described above need to be coordinated in some way so that information presented to the learner is both timely and effective.

2.3 AGENT ARCHITECTURE

Multiple agent tasks *could* be programmed into a single entity, (effectively an expert system with multiple skills) but such a multi-faceted agent would be difficult to construct and maintain. Its very size and complexity would likely affect response times, especially when implemented across a busy network. An alternative design approach is to implement a multi-agent system as described by Bigus & Bigus, (1998) where each modular agent acts more or less autonomously to carry out a specific task. In this scheme, a coordinating agent manages the various agents in its care and passes the results to an interface agent, which communicates directly with the learner.

Researchers such as Inaba (1995) have implemented multi-agent schemes in the past, often in a situation where net-based transactions are occurring. Typically, they include *interface* agents, *informational* agents and *coordinating* or *planning* agents. In extending this concept to include agents specifically designed for a distributed collaborative learning environment, additional entities are proposed, including;

- A LOCAL agent – that has local knowledge, but not specialized knowledge. It acts as a guide or prompt to modules, explaining the actions of various screen objects such as input and result fields and buttons.
- A TRACKER agent – that tracks a user's passage through the program, makes inferences about a user's current level of understanding (i.e. basic user modeling), and keeps a record of a user's subject-specific input. It may also maintain a log of a user's previous sessions for within-course assessment.
- A DICTIONARY agent – that offers contextual explanations/definitions. It may make use of alternative input modes such as audio-based requests, or gestural (deictic) responses (such as circling a word).

The tutor's basic architecture determines its functionality and interactivity with users. In order to enhance collaboration, Genesereth *et al.* (1992) organize agents into two distinct approaches: *direct communication* (where each agent handles its own coordination) and *assisted coordination*, which uses system-level programming to effect coordination. In a variant of the direct communication approach, El-Refai *et al*'s (1997) tour-planning agent system implements the *federated system* approach, where agents communicate with each other via *facilitators* that act as mediators between agents within functional groups. In the proposed approach, a variation on the federated system, detailed in Figure. 1, the coordinating agents fill the facilitator role.

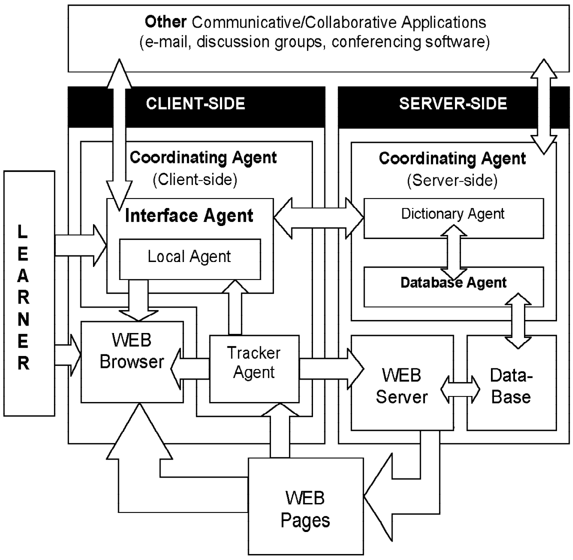

Figure 1: A Proposed Architectural Scheme of a Multi-Agent System

In the figure above, the Database Agent retrieves database information by developing appropriately structured query statements. As the only agent with knowledge of the database and its contents, it mediates requests from other agents such as the Dictionary Agent and system-level requests from the web server. The Tracker Agent has two tasks: (1) it keeps track of the current page so that it can supply the Local Agent with links to information about the current page contents, and (2) it logs a user's entire session, including pages visited and results of any tests or other interactions. While complex tutorial agent activities are currently unimplemented, the data supplied by the Tracker Agent provides the potential for such enhancements to the system in future versions.

A user's interaction is primarily with the Interface Agent, which can interact outside of the web browser window, launching external applications and other windows containing data such as text definitions as needed to enhance the learning experience.

2.4 CURRENT IMPLEMENTATIONS

All agent-assisted learning programs recently developed within our research group make use of Microsoft's ActiveX control technology. Designing courseware for the World Wide Web requires a more flexible approach to delivery. The use of platform-independent shell languages such as Java means better equity of access to online courses without the need for expensive system upgrades or changes. A prototype Java-based multi-agent program for instruction in aspects of medical science currently in development makes use of this flexible delivery approach. An added benefit is that course components can be quickly re-purposed for other disciplines due to the standardized structure.

2.5 MULTIMODAL AGENT REPRESENTATION

A common theme explored by many researchers in pedagogical agent research is the concept of how agents should be represented. Another way of looking at this idea is to ask what mode of representation will aid accomplishment of the learning objectives most effectively. Early programmed instruction courseware employed pop-up text boxes to either admonish or praise a learner. Ubiquitous graphic user interfaces encouraged the development of animated graphic agents, sometimes with a mixture of text-balloons and spoken text-to-speech (TTS) output. Lester *et al*'s (1997) study, described in detail in section 2.6, saw the importance of this aspect of agent design, exploring the concept of agent representation with a rather complex experiment and evaluation.

2.5.1 Communication Modes

Agents can make use of a variety of input and output modes in communicating effectively with a user. Audio-based representations have suffered from lack of development until relatively recently, due mainly to the difficulty and expense of implementing speech recognition (SR) and TTS modes of communication. Recent releases of commercial SR systems that allow continuous speech input, and TTS systems that produce high-level, prosodic voice output have added a range of new possibilities for agent representation and communication. A study by Rudnicky (1993) showed that students tested with a program that allowed any of three different modes of input showed a marked preference for voice (i.e. using SR) over text-field input or choosing from a scrolling field choices, even thought the latter method produced a noticeably faster result. This result could be partly due to the so-called "Hawthorne effect" where the introduction of a new technology produces a momentary increase in production purely because of its novelty value to users. Oviatt's (1995) study showed that multimodal input (combining speech and mouse-pointing input) led to significant improvements in task-completion times in an interactive map setting when compared to single modes of input. The implications for agent representation are that designers need to think about the value of giving their agents additional abilities such as SR and TTS, which may be a preferred mode of user communication in some educational contexts. An obvious application of a non-graphical (i.e. audio-only) agent interface is where the learner needs to give his or her attention to a visual sequence on the screen, while receiving advice or prompting from the agent. Second language learning programs could also benefit from such an agent interface, with the agent providing dialogue cues or help in the target language.

As the purpose of this paper is to provide an overview of multi-agent design, we haven't included a discussion of specific agent implementations. Our experience in designing agent help systems for a number of on-line learning systems does, however, allow us to generalize about aspects of agent design and modes of representation. We have found that a modular agent design gives us a large amount of control over agent development, allowing rapid prototyping of complex systems such as may be required in a collaborative learning environment. Ideal modes of agent representation inevitably depend on the learning context. Consequently, the scope and media content of the material to be learned determines the design of the best mode or modes of both agent representation and communication.

2.6 INNOVATIVE PEDAGOGICAL AGENT DESIGNS

(1) ADELE – An Agent for Distance Learning Environments

Johnson and Shaw (1997) identified factors such as slow access to course materials, courseware that doesn't adapt to individual users, and the difficulty of programming interactivity into learning programs as some of the problem areas common to many web-based learning environments. They developed an Agent for Distance Learning Environments (ADELE) to address problems encountered in the delivery of Web-based courseware. One agent setting was a web-delivered trauma care course where students had to learn a correct procedural sequence for assessing an injured patient. The agent can play example audio tracks such as breath sounds to indicate the current status of the patient. When students deviate from the accepted sequence in some way, ADELE can interrupt and point out that the student isn't following the standard primary survey procedure. The agent can then prompt the student to continue, using the correct procedure. When the procedure is complete, students can query ADELE for further comments, prompting a detailed explanation of parts of the procedure, or steps that should have been taken. Agent tasks are performed at the client side, a decentralized approach which differs from the more common agent architecture which directs most activity from a centralized server. Server-side components include course management, a web server and a database. This system is notable for its use of a client-side intelligent tutor system, which includes a graphical agent.

(2) Herman the Bug

An animated pedagogical agent, Herman the Bug, is implemented in the Design-A-Plant learning environment used in the domain of botanical anatomy and physiology studies at North Carolina State University. Lester *et al.* (1997) introduced the agent as part of a study to investigate the effectiveness of a number of factors involving the use of pedagogical agents in a learning environment. They were interested in questions such as: Do animated pedagogical agents aid or hinder learning? What type of advice should such agents offer (principle-based, task-specific or their combinations)? Which mode of representation (e.g. visual or auditory) or modal combinations should agents adopt to improve problem solving? Their approach was to clone the agent into five versions, each with a distinct mode of interaction, such as animated explanations with audio, audio-only explanations, and task-specific advice about what action the student should take next. Results from the evaluation showed three significant results:

(1) animated pedagogical agents can improve student performance;

(2) agents that provide multiple levels of advice and that use multiple representation modes produce the best problem solving performance; and

(3) complex problems benefit most from pedagogical agent help systems.

The two examples detailed above provide some evidence of the benefits of including multimodal agent or multiagent systems in learning programs. Our own experience in designing learning systems agrees with the findings of both of the above studies, at least at a qualitative level. Future evaluations of our learning programs that include multi-agent help systems will hopefully provide both quantitative and qualitative evidence of the validity of including such systems in on-line courses yet to be built.

3. AGENTS AND FLEXIBLE LEARNING

The distributed collaborative learning approach adopted in the design of courseware developed by this institution uses the concept of flexible delivery as a basis for the design process. Apart from the delivery of course materials over a network, a course typically includes a number of communication modes including student-to-student student-to-lecturer and student-to-agent modes. Currently, all student-agent communication occurs as a client-side interaction, usually through a pedagogical Interface Agent.

Ubiquitous web-browsing engines give us the opportunity to exploit web-wide resources, providing learners with an endless supply of information. Part of our current research is dedicated to devising agent systems that can help a user to access this information in a structured way, by integrating it into the course material. Courseware designers can therefore be more flexible in their choice of course-based resources, allowing users to access new or updated information as soon as it becomes available.

4. CONCLUSION

We believe that multi-agent help systems will provide a means of dealing with the information overload that inevitably accompanies learning programs delivered across a network. Agents will use a variety of communication and representation modes to help us to understand and make use of on-line course materials developed in the near future. This paper has attempted to provide an overview of the likely direction such developments will take. One thing we can be sure of: learning environments employing multi-agents systems are certain to allow both students and courseware developers added flexibility in achieving their learning objectives.

5. REFERENCES

Bigus, Joseph and Bigus, Jennifer (1998). Constructing Intelligent Agents with Java: A Programmer’s Guide to Smarter Applications. John Wiley & Sons, New York. (Introduction: xxx-xxxiv)

El-Refai, M., Kantardzic, M., Elmaghraby A. (1997). An Intelligent Agent-Based Tour Planner. Multimedia Research Laboratory, Speed Scientific School, University of Louisville.

Available from: http://web.spd.louisville.edu/~myelre01/Java/Agents/TourPlanner/papers/Saudi97/saudi97.html. Accessed July, 1998.

Genesereth, M. R. and Fikes, R. (1998). *Knowledge Interchange Format: Version 3.0 Reference Manual Report Logic-92-1*. Computer Science Department, Stanford University, 1992. Accessed July. Available from: ftp://ftp-ksl.stanford.edu/pub/KSL_Reports/KSL-92-86.ps.

Inaba, M. (1995). Internet Consultant: An Integrated Conversational Agent for Internet Exploration. In Proceedings of the Global Information and Software Society Internet Conference (GISSIC’95)

Johnson, W. and Shaw, E. (1997). Using Agents to Overcome Deficiencies in Web-Based Courseware. A paper delivered at the Workshop on Intelligent Educational Systems on the Word Wide Web, Eighth World Conference of the AIED Society, Kobe, Japan.

Lester, J., Converse, S., Stone, B., Kahler, S. and Barlow, T. (1997). Animated Pedagogical Agents and Problem-Solving Effectiveness: A Large-scale Empirical Evaluation. A paper delivered at the Eighth World Conference of the AIED Society, Kobe, Japan.

Oviatt, S. (1995). Multimodal Interfaces for Dynamic Interactive Maps. Dept. of Computer Science and Engineering, Oregon Graduate Institute of Science and Technology. Available from: http://www.cse.ogi.edu/~oviatt/slo_txt.html Accessed February, 1998

Rich, C., and Sidner, C. (1997). Collagen: When Agents Collaborate with People. A paper delivered at the Proceedings of the First International Conference on Autonomous Agents, Marina del Rey, California.

Rudnicky, A. (1993). Mode-preference in a Simple Data-retrieval Task. In *Proceedings of the ARPA workshop on Human Language Technology*, San Mateo. 364-369