The Knowledge Engineering Review, Vol. 25:2, 111–135. © Cambridge University Press, 2010
doi:10.1017/S0269888910000044

Agent-based systems for human learners

ELIZABETH SKLAR[1] and DEBBIE RICHARDS[2]

[1]*Department of Computer and Information Science, Brooklyn College, and Department of Computer Science, The Graduate Center, City University of New York, Brooklyn, NY 11210, USA;* [2]*Department of Computing, Division of Information and Communication Sciences, Macquarie University, Sydney NSW 2109, Australia;*
e-mail: sklar@sci.brooklyn.cuny.edu, richards@ics.mq.edu.au

Abstract

Applying intelligent agent technologies to support human learning activities has been the subject of recent work that reaches across computer science and education disciplines. This article discusses agent-based approaches that have been designed to address a range of pedagogical and/or curricular tasks. Three types of agents are identified in the literature: *pedagogical agents*, *peer-learning agents*, and *demonstrating agents*. Features of each type are considered, as well as the systems in which these agents are incorporated, examining common and divergent goals, system and agent architectures, and evaluation methodologies. Open issues are highlighted, and future directions for this burgeoning interdisciplinary field are suggested.

1 Introduction

Historically, computer systems have been used in many different ways to assist in human learning. In the late 1970s, traditional computer-aided instruction merged with artificial intelligence (AI) to create the field of intelligent tutoring systems (ITS), which exhibited features such as customizing for individual users by tracing problem-solving sessions and responding dynamically (Brown & Burton, 1978; Clancey, 1986). In the 1980s, intelligent tutoring work centered around memory modeling (Schank, 1981; Kolodner, 1983), rule construction (Anderson, 1982), and representation of students' misconceptions (Soloway *et al*., 1981; VanLehn, 1983). Systems developed and tested in the 1990s employed a range of techniques, such as granularity-based reasoning (McCalla & Greer, 1994), Bayesian methods (VanLehn *et al*., 1998), case-based reasoning (Shiri *et al*., 1998) and reinforcement learning (Beck, 1998), to model students and improve their on-line educational experiences. At the same time, early work in the field of *autonomous agents* was beginning to appear (Maes, 1994a; Wooldridge & Jennings, 1995), and it would not be long before the first agents found their way into intelligent tutoring and other types of computer-based systems designed to promote human learning.

Broadly speaking, computer-based systems have been applied in the field of human learning for three different purposes: (1) to replicate human behavior, (2) to model human behavior, or (3) to augment human behavior. The first class of system uses a 'black box' approach in which the goal is to approximate *the outcomes* of human behavior. This class seeks to replicate or replace human activity, cognition or even physiology, to varying extents. Expert systems (also referred to as 'knowledge-based' systems) and some robotic systems fall into this category. Computer scientists and engineers are often focused on building systems of this class. The second class of system takes a 'white box' approach in which the goal is to imitate *the processes* underlying human behavior. This class is developed with the purpose of modeling humans in order to better understand how, and perhaps why, humans act as they do. The aim is not to replace, but to investigate human activity. Research in this area is often driven by social scientists, particularly psychologists or sociologists. The third class of system aims to augment human behavior where the goal is to facilitate *the acquisition of knowledge* within a domain of interest, helping the learner become proficient and gain experience in that area. The aim is to provide artificial partners for joint human activity. These systems are typically developed in collaboration with educationalists and trainers.

Across these three categories, two different perspectives are taken: *passive* and *active*. The *passive* perspective is typically employed by the first two classes of systems, where the human learner is either (1) replaced or (2) studied, but is not (necessarily) a participating player in the system's use. The *active* approach is typically taken by the third class of system, in which the human learner is a participant and thus a direct beneficiary of the system. We note that the three system classes and two perspectives are not mutually exclusive; a single system may employ multiple classes and perspectives. For example, understanding gained from the second class of exploratory system (a model of processes underlying human behavior) may be used to build the first class of system (an agent replacing human behavior), which in turn, may be used as the foundation of an artificial learning peer in the third class of system (augmenting human behavior). Thus, work in the broad area of computer-based human learning systems may bring together techniques and researchers from a wide range of disciplines, united by the common goal of aiding human learners.

The focus of this article is primarily on the third class: facilitating systems that take an active perspective by directly involving the human learner as a user. *In particular, the emphasis here is on implementations that employ intelligent, autonomous agents*. The goal of this article is to highlight ideas and systems representative of the work being developed within the agent-based systems community. The list is not meant to be exhaustive, nor can it be, since new systems are appearing every day. We do not discuss pedagogical issues or learning environments from an education researcher's or developmental psychologist's viewpoint, but rather we focus on design, implementation and evaluation aspects from a technical, computer science, agent-based research, and development viewpoint.

This article is organized as follows. We begin by defining agent-based systems (Section 2), to provide a brief overview for the uninitiated reader and to ensure common terminology for the initiated. Section 3 briefly describes features of computer-based systems for human learning, distinguishing agent-based from other approaches. Section 4 discusses systems designed to support human learning, offering a tripartite categorization of such systems in which intelligent agents are employed. Then, Section 5 discusses system architectures commonly used in agent-based systems for human learning. Section 6 delves into the topic of testing and evaluation, briefly outlining key components of assessment in human learning and evaluation of computer-aided learning systems. Finally, we close by highlighting current open issues.

2 Agent-based systems

Within the field of computer science, the precise definition of an *intelligent agent* is often debated (Wooldridge, 2002), yet there is general agreement that an agent is 'anything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors' (Russell & Norvig, 2002) (see Figure 1). The most fundamental property of any agent is that it is *autonomous*—it decides for itself what to do (Wooldridge, 2002). The agent is equipped with some high-level set of goals, and every time it has a choice of action, it chooses the action that, so far as it knows, is the one that best achieves its goals. This implies that the agent has some internal set of rules guiding its decision-making and cannot be directly told what to do by another agent, or by a human user in an interactive agent-based system. If one agent (or a human user) wishes to change the behavior of another agent, it can only do so indirectly—by making some change that alters the best way for the second agent to achieve its goals. For example, the first agent might convince the second that it is in its best interest to agree to do whatever the first agent is asking at the time.

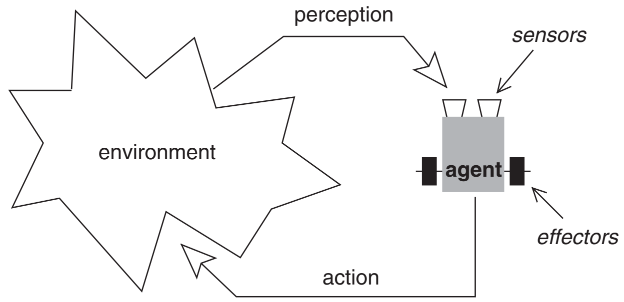

Figure 1 A canonical view of an intelligent agent

Agents maintain a set of *beliefs*, describing the internal model of their state and the environment in which they act. Agents' goals can loosely be thought of as *desires*, things that the agent wishes to bring about; and associated with these desires are various levels of *commitment*. Desires become *intentions* once agents commit to bringing them about, and this typically results in the agent devising a *plan* to achieve them. In the classical view, agent behavior consists of three abstract phases: *sensing*, *planning*, and *acting*. This view stems from seminal work in AI (Arkin, 1998) implemented in the robot Shakey in the 1960s (Nilsson, 1969, 1984). In the first phase, an agent senses its external environment and its internal stance. In the second phase, an agent plans what it should do next, using information that includes sensor data and knowledge of its goals. In the third phase, the agent performs the action selected in the second phase. Early work showed that implementing these three phases as distinct, sequential tasks in an iterative cycle, in what is called a *deliberative* control architecture (Albus *et al.*, 1987), is impractical in dynamic environments and produces unsatisfactory results, particularly with real robots (Brooks, 1984). As a result, *reactive* control architectures (Arkin, 1995) were developed: reactive systems do no planning—agents just sense their environment and follow a set of rules that indicate what actions should be taken in response to the sensed data. The realization that reactive and deliberative approaches do not always work alone, but can be combined to produce superior results (Arkin, 1998, 1989), led to the development of *hybrid*methods. Hybrid models typically use reactive methods to take care of low-level operations, like obstacle avoidance, and deliberative approaches to provide higher-level functionality like planning, joined together with some type of mediation process that coordinates between the two. Hybrid approaches have become dominant within robotics and are often implemented in a *behavior-based* architecture (Arkin, 1998), where low-level actions (such as 'move forward') are grouped together and linked to intensions (such as 'chase ball').

If an agent has a physical body that can move around and interact in the natural, physical world, then we refer to that agent as an *autonomous robot* or simply a *robot*. Agents that exist only as entities within a software system are referred to as *virtual agents*. Note that other definitions of 'robot' exist within the broader, interdisciplinary field of robotics, which includes mechanical devices that are not autonomous, for example, tele-operated machines and industrial 'robot' arms. Some researchers refer to robots as 'embodied agents', but this term is also used to describe simulated agents that emulate human form and motion. In this article, we use the generic term 'agent' primarily to refer to virtual agents and will specify 'robot' when referring to physical, autonomous, intelligent, mobile devices. A *multiagent system (MAS)* is a physical or virtual environment in which multiple agents interact with each other, either directly or indirectly.

In Section 4, we categorize and discuss three primary types of agents employed in human learning systems today, falling into the class of systems that augment human behavior. Primarily, these agents take on active roles, though some are passive in order to make observations before acting. These three types of agents are: *pedagogical* agents that provide overt instruction to learners (Section 4.1), agents that collaborate and act as *peer learners* (Section 4.2), and agents that *demonstrate* aspects of phenomena by interacting with a physical or simulated world (Section 4.3). We note that some of the systems presented as examples in one section could also fit into another section. But first, the next section provides an overview of the components of computer-based human-learning systems, to explain how agent-based approaches fit into traditional work.

3 Human learning systems

The primary goal in any human learning environment is for the learner to advance. Software applications built to facilitate human learning, in contrast to applications constructed with other goals in mind, are not designed to simplify or perform a task for a user, but rather to help a user acquire a set of skills, learn how to accomplish a set of tasks, understand new concepts, learn how to solve problems, and/or practice any or all of the above (Sklar, 2003). These goals are strikingly different from other (i.e., non-educational) interactive agent-based systems where agents are '*assistants*', designed specifically to perform tasks *for* a user. Agents that operate as automated assistants are typically personalized to individual users and historically have addressed a variety of tasks, such as browsing the web (Lieberman, 1995), sorting email, and filtering news group messages (Cypher, 1991; Goldberg *et al.*, 1992; Lashkari *et al.*, 1994; Lang, 1995), or finding other users who share similar interests (Foner, 1997; Kuokka & Harada, 1997; Balabanovic, 1998). Automated assistants are designed to relieve a user's burden by taking over repetitive duties or by streamlining access to complex, overwhelming data sets. In studying and developing these types of systems, the typical kinds of research questions asked center around discovering and measuring ways to make tasks easier for a user to perform. Figure 2a illustrates typical components of a non-educational interactive agent-based system, where agents employ a *user model* (which might be simply a set of user-defined preferences or could be a more sophisticated representation of user behavior such as a Bayesian network) to provide *system adaptivity* by responding dynamically to the needs of the user and/or changes in her environment.

Whereas non-educational interactive agent-based systems are designed to be good assistants, simplifying a user's experiences, systems built for human learning should provide challenges for

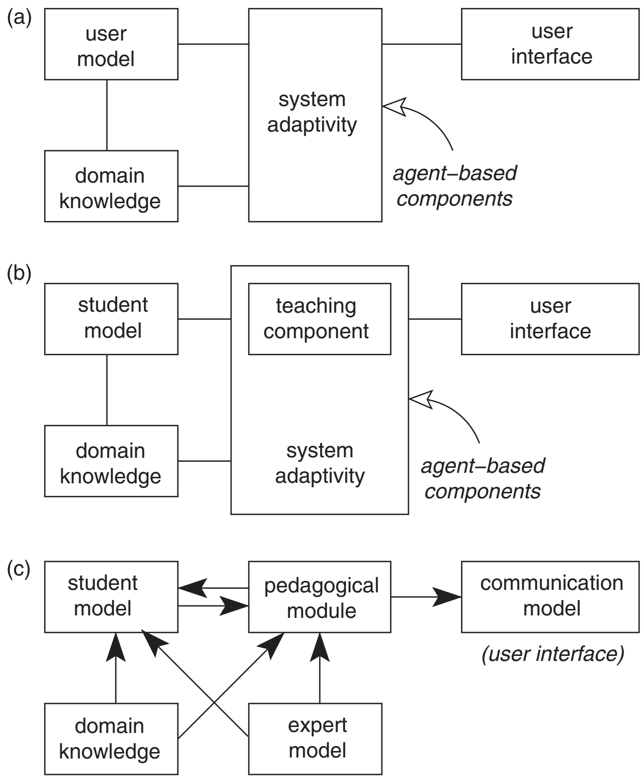

Figure 2 Comparison of interactive system components: (a) interactive agent-based system, (b) interactive agent-based learning system (adapted from Sklar & Richards, 2006), and (c) intelligent tutoring system (from Beck *et al.*, 1996)

the user. The overriding system goal is for the user to learn how to perform a given task, so the system should make the process of learning how to accomplish that task easy—the process of learning the task, not the task itself (Sklar, 2003). This viewpoint was elegantly expressed by (Malone, 1981a, 1981b) within the context of computer games. He makes an important distinction between 'toys' and 'tools', defining toys to be systems that exist for their own sake, with no external goals, and tools to be systems that exist because of their external goals. Good tools should be easy to use, in order to expedite the user's external goal. Good games should be difficult to play, in order to increase the challenge provided to the player. Interactive learning systems should also increase the challenge, to keep learners engaged in a continuous learning process.

The general term *interactive learning system* (ILS) includes not only the more specific (and perhaps more familiar) term ITS, but also provides a broader definition encompassing environments that are designed for more exploration on the part of the student than ITSs (which are typically more structured and scripted according to carefully engineered, domain-dependent models). A typical ILS includes the following components (Mark & Greer, 1993; Sklar, 2000):

- *domain knowledge*: a representation of the topic that the student is learning;
- *teaching component*: an instructional model that is used to guide the student through the knowledge domain;
- *user interface*: the interaction mechanism that lies between the human student and the computerized system;
- *student knowledge*: a 'user model' of the student in relation to the domain knowledge, indicating how much of and how well the student knows the domain; and
- *system adaptivity*: the means by which the system adapts automatically to the student's behavior, backtracking when the student makes mistakes and moving ahead when the student demonstrates proficiency with portions of the domain.

Figure 2b illustrates these components and their relationships to each other. This structure is quite similar to that of a non-educational interactive agent-based system, such as that shown in Figure 2a; the *student model* provides a function similar to that of the *user model* and the *teaching component* is added to enhance *system adaptivity* for the changing needs of the learner. In both cases (educational and non-educational systems), different types of agent technology can be included at different levels. For example, where interface agents are used in non-learning applications to help a user navigate a complex domain, pedagogical agents are implemented in a learning application to guide users through a problem space. Understanding and embracing the special characteristics that learning tasks require of an application is necessary in order to adapt techniques from non-learning applications to those designed for education and training. Figure 2c illustrates the traditional components in an ITS (Beck *et al*., 1996). The main difference is the use of a specific *expert model* that works in tandem with a *pedagogical model* to select learning experiences for the student, often from a predefined database of cases describing common student misconceptions. In an agent-based learning system, the *teaching component* is embedded in *system adaptivity* and can often adapt while the system runs. The key difference is that in a traditional ITS, system control is essentially 'deliberative', whereas in an agent-based environment, system control is essentially 'behavior-based'.

4 Types of agent-based systems for human learning

Within the class of facilitating agent-based human learning systems, we have found that there are three primary types of agents. The first are pedagogical agents (Johnson, 1995), personalized assistants that interact directly with a learner and explicitly guide her through the domain. Referring back to Figure 2b, pedagogical agents are most overtly involved in the *teaching component* and *user interface*. Typically they consult the *student model* in order to understand the learner and provide feedback that encourages the learner within her appropriate 'zone of proximal development' (Vygotsky, 1978). Pedagogical agents usually have full access to the *domain knowledge*; i.e., they know all the 'right answers', though depending on the teaching style that is embodied, how and when they share their knowledge with the student varies. They provide *system adaptivity* by responding to the student's needs dynamically. In immersive implementations, they also react to changes in the environment. The second type of agents are peer-learning agents, interactive partners in the learning process itself (Kim, 2005; Sehaba & Estraillier, 2005). These agents are built into the *user interface* and, as with pedagogical agents, have knowledge of the user (through the *student model*). However, they do not have complete access to the *domain knowledge*; as peer learners, they have similar grasp of the knowledge being acquired as the student. While these agents may have teaching capabilities (implemented in a *teaching component*), their instructional behaviors are much less engineered for guiding learning overtly than pedagogical agents. For example, educational or training games are a common type of scenario for these type of agents, where they act as players alongside the human learners. These agents provide *system adaptivity* by reacting dynamically to both the student and the environment (such as a game). The third type of agents are demonstrating agents, where the agents themselves are interactive mediums for learning, for example, agent-based simulations (Repenning & Citrin, 1993) or educational robotics (Goldman *et al.*, 2004; Sklar *et al.*, 2005). These agents embody the *domain knowledge* and are removed from the other components of the learning system (i.e., there is no *teaching component* or *student model*). In a way, these agents *are* the *user interface*. Any *system adaptivity* is put in by the user, i.e., in creating different programs for a robot. An open area for future work is the development of systems that combine this third type of agent with one or both of the others.

4.1 Pedagogical agents

Much like a narrator in a movie who provides voice-over to explain scenes but never actually appears in the film, a pedagogical agent 'pops up' when the learner indicates (directly or indirectly) that she needs help. There are a number of design issues in constructing this type of agent. The first is *appearance*, deciding what the agent will look like and how it will interact with the user. An agent might be animated and engage with the user through natural language conversation. A more primitive agent might appear as simply an icon and provide feedback to the user through text bubbles. Animated agents offer the opportunity for more engaging and immersive experiences for the user, though these naturally require more development time. A second design issue is *initiative*, determining when the agent should become visible to the user. The question of initiative is important in any interactive system, regardless of whether it is educational or agent-based. There are generally three methods used to determine when an agent should appear to a user: *directly*, upon request by the user, for example by clicking a 'help' button; *indirectly*, by the system monitoring the learner's performance and automatically detecting when she seems to need assistance; and *mixed initiative*, which relies on a combination of the first two. With most agent-based systems for human learning, the second method is used. The third design issue is *purpose*, identifying the reason(s) for using agents in a system. The earliest agent-based systems for human learning were designed in response to issues in traditional ITSs, providing a means for the 'tutor' to be an integrated participant in the learner's experience (Johnson, 1995). The purpose of these systems was to provide pedagogical assistance with academic subjects or professional development, such as military training. More recent systems broaden the areas of application to include organizing and monitoring access to course materials, providing advice on degree planning, and facilitating group learning. In this section, we offer examples of systems that focus on each of these issues.

Johnson (2001) presents a comprehensive review of his research on *animated pedagogical agents* (APAs), highlighting issues that are faced when developing computer-based learning companions. Questions are addressed such as when to initiate dialog, how to include non-verbal communication, and how to design for a wide range of client technologies. Animated agents can mimic human gestures and emotion-driven expressions, which can help engage students more than static avatars. Several agents were developed and tested, including: Soar Training Expert for Virtual Environments (STEVE) (Rickel & Johnson, 1998), Agent for Distance Learning, Light Edition (Adele) and Herman the Bug. STEVE was constructed for adult learners, embedded in an immersive three-dimensional environment that included sound and was designed for training naval personnel. Adele was built for university students, to be accessed over the Internet using standard workstation platforms and web technology. Herman the Bug was created for young students, a 'talkative' agent that gave advice about plants while teaching about botany. Animated agents can provide a number of advantages, such as navigational guidance, interactive demonstrations, gesturing for attention, and use of other non-verbal communication devices. Empirical results using APAs are positive, with some cautions. For example, there is evidence that young students may be distracted from the learning task by the agent's interface; however, the authors believe that a well-designed agent can keep students on task and avoid this pitfall.

In a more recent example of an application built for adult learners, Blake *et al.* (2007) describe an 'intelligent scaffolding agent' for use in human training environments. The role of the agent is to devise an appropriate level of assistance for the human in order to enhance the performance of the learner. At first, the agent provides multiple 'training aids', but gradually removes these as the learner acquires skill with her assigned tasks. In this system, the *domain knowledge* is organized into specific scaffolded levels, and the agent consults the *student model* to determine which level should be presented to the learner at any given time. This architecture combines the advantages of traditional ITSs, where the tutor's actions are carefully engineered, with agent-based systems, where the tutoring agent's actions are behavior-driven, by engineering the portion of domain knowledge accessed by the agent, adjusting that portion dynamically as the learner advances and implementing the agent that interfaces with the learner using behavior-based methods.

Much of the work involving APAs focuses on interactive *pedagogical drama* (Marsella *et al.*, 2000), where the agents become actors in a pseudo theatrical environment and learners either become immersed as participants in the drama or act as observers, like members of an audience. There are advantages to each approach; the former approach requires learners to act in the drama, which can be challenging and motivating, while the latter gives learners opportunities for reflection and perhaps impartial analysis. Fassbender and Richards (2006) have designed VirSchool to allow a student to explore a topic area, such as philosophy, facilitated as an adventure-style quest. Johnson (2001) and Marsella *et al.* (2000) use an interactive pedagogical drama for a system called 'Carmen's Bright IDEAS' in which an adult human is guided through scenarios designed to improve problem-solving skills. Carmen, a character in the drama, is the mother of a pediatric cancer patient; she has a job and has another (healthy) child to mind. When using the system, the learner observes Carmen's thoughts and can choose actions for her, in sessions with a counselor, discussions with her child's doctor, interactions with her boss, and so on. The pedagogical goal is for the human user of the system to improve her problem-solving skills and gain insight into similar situations in her own life. The underlying architecture of Carmen's Bright IDEAS has been developed into a generalized framework called Thespian (Si *et al.*, 2005a, 2005b) and applied to other domains. The Tactical Language Training System (TactLang) (Johnson *et al.*, 2004) is a military language training system in which the learner engages in role-playing activities to acquire knowledge of the language, idiom and customs of particular geographic regions. One example is a drama, which unfolds in a village cafe in the Middle East, and the learner interacts with characters who are speaking Arabic. The system architecture integrates PsychSim, a multiagent system for mental modeling (Pynadath & Marsella, 2005), with a storyline and basic script as well as pedagogical goals and social norms to guide agent–agent and agent–human interactions. The premise is that this type of modular system architecture could aid in rapid deployment of other interactive pedagogical dramas applied to other domains, by plugging in different *domain knowledge* modules (i.e., scripts) to be accessed by the more generic behavior-based agents.

The Internet has opened up opportunities to explore the application of agent-based systems designed specifically for human *distance learners*. APAs have been implemented in these types of systems as well. For example, Lucas *et al.* (2005) define APAs in their MAIDE (Modelagem de Ambientes Inteligentes de Aprendizagem) system to assist students in distance learning environments. An APA perceives that the user needs assistance based on a model of the learner and the learning goals of the system. The agents pop up and provide guidance to the learner regarding the use of a calculator. The APAs work in conjunction with the initial loader (instantiated by the Foundation for

Intelligent Physical Agents (FIPA) Open Source (OS) Agent-Loader) and a graphical interface agent called the CalcAgent for handling the display, input, and output to the calculator. Examples of other systems designed to support distance learning in a variety of ways are described in the next few paragraphs.

Some systems combine the first and third classes of agent-based systems for human learning, as described in Section 1, by designing a pedagogical agent's behavior to *replicate* a human expert from whom a novice can learn. One example is the Online Mediator Education System (Tanaka *et al*., 2006), where students are able to practice their legal negotiation skills with the assistance of a mediator agent. Within the legal profession, a technique known as alternative disputation resolution (ADR) is becoming a common alternative to courtroom trials. To become competent in ADR, law students typically undergo training with an experienced mediator. However, this is costly and time-consuming. An advice agent that embodies the role of a mediator was created by Tanaka *et al*. (2006). Typical of systems in law-related fields, a case-base (i.e., a database of cases) is used to find examples, together with a mediation model, issue points, similarity measure, and rule-based advice model. The advice agent uses the cases and models to provide guidance and evaluation to trainees. It appears on the screen as a talking head, complete with facial expressions and speech with intonation. Preliminary evaluation studies show that about half of the recommendations given by the on-line case-based training system were accepted by the participants. The developers note that more cases and rules are needed to improve this outcome, eventually leading to a system that can provide recommendations directly to the affected parties.

A growing number of systems have been developed to support a range of distance-learning needs that are not directly focused on acquisition of domain knowledge, but rather on aspects such as organizing and monitoring access to course materials. Teachers face challenges of monitoring progress and sustaining morale when students participate with a learning system remotely. These problems are exacerbated when there are large numbers of students for a teacher to monitor. One example of a system that seeks to address these issues is the agent-based intelligent tutoring system (ABITS) (Mowlds *et al*., 2006), which instantiates rule-enhanced agents guided by a *belief desire intention (BDI)* architecture (Bratman, 1987) to identify personal preferences, learner type, and learning style. An individualized *student model* is used in conjunction with an agent who is assigned to each student and assists the student in achieving his/her goals. The agent observes the student's interaction with the e-learning system to formulate its beliefs about the student's level of involvement with the online course. The student's learning style is also added to the agent's set of beliefs via analysis of the responses to a learning style questionnaire. Support provided by ABITS includes: weekly emails to the student summarizing what s/he has achieved, with the aim of providing positive feedback and reinforcement; reminders from the agent regarding due dates for assessments, which require acknowledgement of receipt from the student; and tailored quizzes and recommended reading material. To further support the ability to suggest appropriate course material, ABITS also offers an alterative architecture that includes machine learning agents that can adapt the system based on observing student behavior. For example, the researchers observed that students typically used the weekly lesson title or sub-title descriptions as the source of keywords to search for relevant material. As a consequence, researchers built adaptive agents to automate this search and recommend further reading for the students to undertake.

Most students typically require assistance with more than just the curricular content and learning materials of one particular class—most students also need academic advising to guide them through their entire degree. Getting timely access and advice is a challenge particularly faced by distributed and remote students. This problem has been addressed using agent technology in the 'e-Advisor' system (Lin *et al*., 2006, 2007) which allows masters students to perform initial and opportunistic planning, that is, dynamic planning performed as needed, to develop a personalized educational program. A number of knowledge models are used, including: ontologies, program regulations, course models, student models, prerequisite relations, and a preference-based optimization model to enable selection of the best plan from a range of plans. Notification, planning, interface, evaluation, and monitoring agents are provided to form a modular multiagent system.

Another open area of research in distance learning environments examines ways to facilitate group-based and collaborative learning where students participate from physically distributed locations. The introduction of an agent can help to play the role of a teacher or group guide. One example of a system that uses an agent-based architecture to facilitate a computer supported collaborative learning environment is Intelligent Multiagent Infrastructure for Distributed Systems in Education (Khandaker & Soh, 2007). In this system, agents are instantiated to represent individual students and a teacher, while a 'group agent' enables communication and group activity. The teacher agent performs some analysis of student agent actions in order to advise the human instructor about how to respond.

This section has highlighted a number of issues in the design and implementation of pedagogical agents, describing examples to illustrate each. The modes of interaction and agent appearance differ, with more developed systems leaning toward the inclusion of APAs that employ natural language methods and gestures for conversing with human learners. Authors acknowledge the labor involved in constructing these types of agents and strive to capitalize on behavior-based agent architectures in order to implement modular systems in which some components can be re-used when knowledge domains change. The audiences for systems involving pedagogical agents vary widely, from young students in school classrooms to adult learners in business or military settings. Distance learning is a growing area of application, where the community has recognized that needs are not only academic but also organizational.

4.2 Peer-learning agents

There are many learning environments where agents interact with human learners as peers. These agents appear less intrusive than pedagogical agents. Many agent-based learning systems leverage game technology to provide both motivation and a situated, simulated training environment (Lave & Wenger, 1991). Human learners engage with agents as opponents or partners. These agents act more like peers than pedagogical agents, which are more like tutors or instructors. The primary issues faced when constructing peer-learning agents are: developing realistic situated environments in which the human learner and agent peer interact, offering believable agents, and providing natural modes of interaction.

To address the issue of developing situated environments, which can become very expensive, many projects take advantage of a number of free or off-the-shelf game engines. For example, the TactLang mission environment (mentioned earlier) employs a modified version of the game engine Unreal Tournament, known as Gamebots (Adobbati *et al.*, 2001). Many games include simulation components (Aldrich, 2003) that provide practical experience and game elements that offer an environment of engagement, discovery, and competition (Bartles, 2003). Simulations can be particularly useful for providing training in several categories of skills: internalizing processes, understanding systems, decision making, perspective shifting, team building, and cooperation (Galarneau, 2004). Wilkinson (2002) stresses the importance of realistic simulation settings for learners, where errors can be expected and the experience of failure helps learners progress in a safe environment. The notion of safety is particularly critical for the mission rehearsal exercise (MRE) project (Swartout *et al.*, 2001). Safety is also a key consideration in the training simulation being developed by Richards *et al.* (2005, 2007) where agents are created in a game environment to allow the user to explore various risk scenarios. This project is focused on addressing issues concerning the agents' acquisition and reuse of knowledge and the language, cognitive, and behavioral abilities of the agents to provide a more believable, engaging, and immersive learning environment. The more recent work on *Airport World* (Richards *et al.*, 2007) has been used to train customs officers employed at airports to identify high-risk situations. The game setting is similar to that of TactLang in which users interact with virtual (animated) characters in an immersive, virtual reality environment.

A number of development toolkits and game environments including programmable game engines are available, each with various strengths and weaknesses. A good overview and list of game engines is provided by Isakovic (Johnson & Onwuegbuzie, 2004). A shorter discussion is provided in Barles *et al.* (2005). Game engines often include: a rendering engine to output 2D or 3D graphics, an animation package, a physics engine (or other functionality to handle object collision), a sound synthesizer and a scripting language. Some incorporate more sophisticated tools and techniques such as networking, threading, scene graphs, and ideas from AI such as agent-based behaviors, rule bases, and knowledge bases. The developer may be able to choose functionality from a range of options. For instance, Garage Games[1] provides tools for creating 2D, 3D, and console games using the Torque game builder and game engine together with academic resources useful for teaching games development. Different types of games can be developed depending on the game engine, for example Torque supports development of First Person Shooter (FPS) Games. Neverwinternights[2] allows game development, using the Aurora Toolset and game engine, of third person perspective computer role-playing games. Neverwinternights is a dungeons and dragons fantasy game which was the first massively multiplayer online RPG (MMORPG). Now eclipsed by games such as World of Warcraft, it remains quite revolutionary in allowing users to host their own MMORPG server and to create their own worlds and adventures with up to 64 of their friends[3]. Middleware such as the general purpose Gamebryo[4] System Development Kit has allowed the development of robust and complete games and even supported the development of more customized and purpose-built toolkits such as The Elder Scrolls IV: Oblivion which use the gamebryo engine[5]. Similarly the IdTech Engine, developed by Id Software in association with Valve[6] and launched in the first FPS Wolftenstein 3D in 1991 (ID Software, 2009), has been an integral part of successful games such as Call of Duty, Soldier of Fortune, Half-Life, Medal of Honor: Allied Assault, Star Trek: Elite Force, Heretic, Hexen, DOOM, and QUAKE. Simulation games have also been used in business environments, for example, in teaching administrative skills.

Game engines have been employed to incorporate the practice of storytelling to guide the human learning process within an agent-based system. In Richards *et al.* (2006), the narrative engine is used to interpret user actions into narrative terms that enrich the experience. Through a decomposition and recomposition process, the user may choose from a wide range of options that can be explored in parallel or sequentially. The narrative engine is used in conjunction with the Unreal Tournament game engine to provide training to customs officers. In contrast, Mott *et al.* (2006) use the hierarchical task network planner technique to determine the order and nature of events together with the game engine HalfLife to develop the 'Crystal Island' learning environment, where the student becomes a medical detective as she explores an island and learns about disease.

To address the issue of providing believable agents, one strategy currently being explored by many comes from examining human characteristics, such as emotion and empathy, and extending agents to handle and even emulate such traits. van den Broek (2005) developed empathic technology agents that mimic human empathy in a study concerning stress levels in individuals, where stress was detected through analysis of human voice recordings. In this study, the goal was not to use agents to teach the subjects, like pedagogical agents would, but rather to add human abilities to the agent controller so that the results would be more sociologically valid, a key concern and common shortcoming of laboratory-style testing with humans. In addition, related to better understanding humans is the work of Sehaba and Estrailier (2005), who use an agent approach to help rehabilitate children with autism. A multiagent system is used to model the knowledge of therapists, the child's profile and the dynamics of the interactions between the therapists and the child. Jarrold (2007) conducted a comprehensive study of autistic behaviors with human subjects and used the results to

[1] `http://www.garagegames.com/`

[2] `http://nwn.bioware.com/`

[3] `http://nwn.bioware.com/about/description.html`

[4] `http://www.emergent.net/`

[5] `http://www.elderscrolls.com/games/oblivion_verview.htm`

[6] `http://www.valvesoftware.com/`

build and test a rule-based system to mimic the responses of autistic children. Spoelstra and Sklar (2007) built a simulation of human learners in groups, as part of the *SimEd* project (Sklar *et al.*, 2004). Their model was constructed from examination of a wide range of pedagogical literature on human learning environments and interactions in group learning. They compared the results of different group compositions with varying the number of 'high' and 'low' ability learners in a group, as well as group size, and the presence or absence of group rewards in learning. Models of emotion were incorporated to reflect the learner's responses to easy and hard lessons.

A key difference between agent systems for human learning and other agent systems is the need for human communication languages as well as agent communication languages (ACL). In general, it is unrealistic to expect a human to communicate with a software agent using a standard (e.g., FIPA[7]-compliant) ACL. The burden is on the system developer to create a communication channel that bridges the human-agent gap. It is common to use a personal agent to bridge this gap. Lazzari *et al.* (2005) use personal agents in their remote assistant for programmers (RAP) system to allow human users to interact with other humans and other parts of the system, each represented by other types of agents. Personal agents are created for each online and offline user. Personal agents handle a range of tasks such as selecting answer types, submitting queries, finding answers, finding experts, receiving expert ratings, selecting experts, receiving answers, and rating answers. Some of these tasks are performed in conjunction with other agents in the system. As in the case of RAP, personal agents are tailored to the particular human user based on a user profile, stereotype or model, and often require a high degree of sophistication. This is clearly true in the case of the user observation agent (UOA) employed in the studies of the behavior of autistic children by Sehaba and Estrailier (Sehaba & Estraillier, 2005). This agent takes input from an integrated software and hardware system known as FaceLab to capture features of the human subject's face and orientation of gaze. In addition, the UOA takes into account actions with the mouse, touch screen, and keyboard.

Kim (2005) uses agents as learning companions, motivated by the pedagogical strategy of providing a learner with peer support. This work reveals that the competency of the learner has a large impact on the nature of the interaction with agents as learning companions, referred to as 'PALs'. Strong students, identified by their grade point average, preferred for the PALs to take a leading role and expected to be given correct advice. Weak students preferred to control the PAL and asked for assistance only when they wanted it and were satisfied with some wrong answers, as they found that a PAL that was always correct was intimidating. These interesting findings emphasize the varied influence of agents for human-based learning. Not only will differences in settings and interfaces affect learners, but also the personalities of the learners themselves will be a factor.

This section has highlighted three issues in the design and implementation of peer-learning agents, providing illustrative examples. This application area is largely dominated by game settings, where the agent peer and human learners interact as collaborative or competitive players. The main difference between a pedagogical agent and a peer learner is that the pedagogical agent acts much more like an overt teacher, monitoring students' actions, and providing instructional assistance when observing that the student has made mistakes or is becoming lost; whereas, the peer learner acts more like a classmate, learning and making mistakes along with the student. An interesting area of research is examining the effectiveness of different reasoning models within peer learners to understand whether students respond better to peer agents that appear more or less intelligent. Some systems integrate peer and pedagogical agents, providing a peer agent that makes mistakes and a pedagogical agent that offers corrections, allowing the human learner to observe and learn from the mistakes of the peer. Researchers are finding that believability of the peer agent is important, and many are studying ways to incorporate emotion models in these agents so that they can express satisfaction, frustration, and other emotions throughout their shared learning process.

[7] Foundation for Intelligent Physical Agents, `http://www.fipa.org`

4.3 Demonstrating agents

The notion of agency is useful for teaching and *demonstrating* a wide range of phenomena in the world. From introductory programming concepts (Blank *et al*., 2005) to dynamic systems (Wilensky & Resnick, 1999), innovative researchers and teachers have developed many different types of agents that interact with students of all ages. These environments take advantage of the popular and proven *constructionist* (Papert, 1991) pedagogical paradigm, motivating students and helping them to learn by doing. Here we highlight two complementary directions within this paradigm that concentrate on the use of agency to provide meaningful learning experiences: *multiagent simulation* and *educational robotics*. These types of *demonstrating agents* are very different from the types of pedagogical and peer-learning agents described in the previous two sections. There is no explicit model of the teacher or the student. The student acts as a programmer, developing her own methods of controlling the agents. In these systems, the agents are the *manipulatives*: 'objects that can be touched and moved by students to introduce or reinforce a [mathematical] concept' (Hartshorn & Boren, 1990), which have been shown to be effective in mathematics education (Suydam & Higgins, 1977).

Educational agent-based and multiagent simulation systems allow students to program agents using simple commands and to view graphically, and instantly, the effects of their code. This not only teaches students about programming concepts, but also provides powerful lessons in modeling. Students observe the world, invent rules about it, program the rules and analyze how well their rules represent the phenomena in the world that they are attempting to model. One of the more widely used agent-based simulation environments in education settings is NetLogo[8] (Wilensky, 2002), developed by Uri Wilensky and others at Northwestern University (Sklar, 2007). Branching out from StarLogo (Resnick, 1997), NetLogo provides participatory modes of interaction. Both StarLogo and NetLogo are based on Seymour Papert's LOGO environment (Feurzeig *et al*., 1970) in which novice programmers control a 'turtle' by giving it simple commands such as 'go forward'. Sengupta and Wilensky (2005) use NetLogo to assist physics students to better understand the field of electromagnetics at the micro level. By modeling concepts such as electrons and atoms as agents, students are able to discover emergent phenomena for themselves and learn to predict behaviors of or within a system in ways that result in deeper learning and understanding. The NetLogo Investigations in Electromagnetics study demonstrates how learning in the domain can be broken down. In this particular domain, 'thinking in levels using multiagent based models allows the students to establish concrete relationships between submicroscopic objects (e.g., electrons) which are shrouded in mathematical equations in traditional physics instruction' (Sengupta & Wilensky, 2005). Blikstein and Wilensky (2005) also employed the NetLogo environment as a way of helping students to understand some of the difficult concepts involved in Materials Science. The MaterialSim system is a modeling for understanding framework that uses multiagent modeling languages, in which each agent is a basic computational construct with simple rules which control their behavior and from which more complex higher level behaviors emerge. The study included classroom observations, pre- and post-interviews and data analysis of the usage session.

REAL (Bai & Black, 2006; Bai *et al*., 2007) is a framework that uses a multiagent system comprising a reflective agent, pedagogical agent, expert agent, and communication agent to model the human roles of user, teacher, domain expert, and a coordinator, respectively. The REAL framework includes a simulated gaming environment that allows learners to reflect on what they know by exploring their own ideas. Users specify behaviors for agents in the system via propositional networks or procedural rules. (Bai & Black, 2006) want to go beyond an educational game system, which may be engaging but not necessarily result in constructive reasoning. The REAL system allows students to debug their own thinking processes by looking at their thoughts represented in the system from different perspectives. The system can incorporate machine-learning techniques such as Bayesian

[8] `http://ccl.northwestern.edu/netlogo`

networks and concept maps. The agents are implemented using a BDI architecture. The work differs from NetLogo in the use of a game environment and the visualization of propositional statements.

Examples of other multiagent simulation environments include Swarm[9] (Epstein & Axtell, 1996), RePast[10] (North *et al.*, 2006) and AgentSheets[11] (Repenning & Citrin, 1993). Only AgentSheets was constructed explicitly for use in education settings, where students give agents behaviors using a menu-driven interface. Swarm, based on the simple notion of ants gathering sugar in a nest, and RePast, a Java implementation with Swarm-like properties, have been quite widely used to model, test and refine a broad range of complex economic, social, computational, and scientific theories. Both of these environments require skilled programmers to implement and control the behaviors of agents, and so are not as accessible in most educational settings.

Educational robotics refers to the use of robots in classrooms to teach a wide variety of topics, not necessarily robotics in particular (Sklar & Parsons, 2002). With the advent of the LEGO Mindstorms Robotics Invention System[12] in the late 1990s, today robots are being used all over the world to engage students from early primary through undergraduate classrooms (Klassner & Anderson, 2003; Blank *et al.*, 2005; Sklar *et al.*, 2005). As outlined in Sklar *et al.* (2005), some university courses focus more on hardware and engineering design aspects of robotics, while others concentrate on control mechanisms, agency, behavior-based paradigms, and multiagent systems. A broad range of experience reports have been published detailing lessons learned using robotics with younger students, in primary and secondary school classrooms and after-school programs (e.g., Martin, 1994; Wagner, 1999; Goldman *et al.*, 2004).

As well as hands-on, hardware-based approaches, a number of simulators have been developed to give students who do not have access to robot hardware an opportunity to explore the concepts behind controlling robots or to speed up development by providing a rapid-prototyping environment where debugging can occur more quickly than on real robots. Chu *et al.* (2005) developed RoboXAP, an agent-based simulation environment for children, designed to be used in conjunction with the popular RoboLab (Erwin *et al.*, 2000) graphical programming interface and LEGO Mindstorms robot. The motivation was to give students an opportunity to learn about agent-based programming by using RoboLab in a 'safe and friendly' place—they can 'try out' programs in the simulator before loading them onto the robot platform and before being faced with real-world, physical constraints, and issues such as noise.

When dealing with varying robotic platforms, even with a relatively simple one like the LEGO Mindstorms, it is effective for students to be able to classify and specify behavior patterns for their robots. Behavior patterns can range from basic actions, such as 'move forward for 4 s', to complex activities such as 'open a gate'. Owing to the complexity of defining and implementing robot behaviors, many control architectures are designed to build complex behaviors out of simple, low-level commands (Mataric, 1998). Another approach by (Goldman, 2005) provides a custom behavior-based interface to RoboLab and an XML-based translator for downloading on to the Sony AIBO robot. Azhar *et al.* (2006) describe an agent-oriented behavior-based interface framework designed to enable learners to specify introductory programming concepts at various levels of abstraction, across multiple platforms within a simulation environment that can be used for testing ideas and programs developed.

The work of Holz *et al.* (2006) combines virtual reality with hardware to create a Mixed Reality Agent. A museum guide was created that combines a virtual agent displayed on a computer screen sitting on top of a physical agent robot. The physical agent allows the museum guide to move around a building, while the virtual agent allows the robot to have a persona that is expressive and adaptable. For example, the interface can be adapted to the type of user, children or adults, or the

[9] `http://www.swarm.org`

[10] `http://repast.sourceforge.net`

[11] `http://agentsheets.com`

[12] `http://www.legomindstorms.com`

virtual reality can be adapted to content where a lion may introduce a display of carnivore skulls or the face of the Mona Lisa may introduce some Italian paintings. BDI reasoning is used to determine where the robot should move to and also which avatar and animation is appropriate.

A new type of mixed interface system that combines demonstrating agents with human learners in an interactive drama is I-Shadows (Brisson *et al.*, 2007). In I-Shadows, an agent-based system creates virtual characters that are displayed on a screen and children interact with the virtual agents using physical shadow puppets. Both physical and virtual worlds exist side-by-side, with events that occur in the physical world being reported to the virtual world, and vice versa. Individual agents take on roles in the puppet drama (such as 'hero' and 'villian'), and each agent is instantiated using the same architecture but with different parameters.

This section has presented two approaches to the use of *demonstrating agents* in educational settings: agent-based simulation and robotics. While there are numerous examples of each type of system implemented in non-educational settings, the last 10 years has shown tremendous growth in these types of environments designed particularly for use in classrooms and other human learning situations. Probably the most critical design issue when developing demonstrating agents is the identification of the audience and thoughtful analysis of the skills of learners before and after engaging with the system. In most cases, learners are novice programmers so the system has to provide subtle lessons in introductory programming in order for the students to control the agents in desired ways—even if the domain knowledge for which the system was built is not programming. The modes of interaction between human learner and demonstrating agent differ, being either virtual, physical, or mixed (as in I-Shadows). Some systems involve programming multiple agents at a time (as in NetLogo), while others focus on single agents (as do most educational robotics systems). In systems with a virtual component, such as NetLogo and I-Shadows, the environment is constrained and the programmer (i.e., human learner) does not have to worry about external factors that might interfere with agents' behaviors, whereas in a physical environment, learners have to cope with issues such as noise and battery life. Systems incorporating demonstrating agents tend to stand apart from the systems discussed earlier in which pedagogical or peer-learning agents are implemented. While some systems employ simulation to create a believable environment for human learners (e.g., Swartout *et al.*, 2001; Richards *et al.*, 2005), these are not truly hybrid systems because the human learners do not program controllers for the agents in the simulation. An open area of research is the development of systems where human learners can program demonstrating agents while also interacting with pedagogical and/or peer-learning agents for assistance in the learning process.

5 System architecture comparison

System architectures can vary widely depending on the purpose of the system, preferences of programmers and designers, and technical practicalities related to implementation. From a general software engineering perspective, traditional system architectural 'design patterns' (Gamma *et al.*, 1995) include: data abstraction, communicating processes, implicit invocation, repository, interpreter and layered (Shaw, 1996), and client-server or peer-to-peer. From the systems highlighted in this paper, we identify three architectural design patterns: *single agent, multiple heterogenous agents*, and *multiple homogeneous agents*.

In the *single agent* approach, one agent possesses multiple capabilities and comprises all the components illustrated in Figure 2b. This pattern is probably the most traditional and could be used to describe early systems before multiagent approaches gained prominence. However, this pattern is not limited to traditional systems and its use is not uncommon today. The single 'agent' approach can be used loosely, to refer to the overall purpose of the system, i.e., an *agent*helping a human learn; or can be more tightly connected to the use of classic agent architectures, such as BDI. Some systems allow instantiation of multiple instances of a single agent, in order to support multiple students working together at the same time, for example STEVE (Rickel & Johnson, 1998). We still label these systems as using the 'single agent' approach, however, since they are primarily designed to operate in a single agent environment, whereas the multiple homogenous systems (described below) instantiate agents that cannot accomplish the system goal alone.

In the *multiple heterogenous agents* approach, multiple agents each have their own goals and tasks to perform and can work together to achieve larger system-level goals. Various architectural design patterns for multiagent coordination, such as broker, embassy, monitor, and mediator are discussed in Hayden *et al.* (1999). We include a *student agent*, as many of the systems reviewed include an agent that explicitly represents (models) the learner. A very commonly used design pattern is to divide the system into multiple components (as in Figure 2b) and have each be handled by a specialized agent. The popularity of this pattern is not surprising as the architecture provides a modular way of decomposing a system into small, manageable components. This pattern achieves the software engineering design goals of low coupling and high cohesion (Sommerville, 1992) by using independent and event-driven software modules in the form of different agents. An example is the dynamic adaptive learning system (Sun *et al.*, 2005) that uses a student agent, an evaluation agent, a record agent, a learning object agent, and a modeling agent. In this system, the agents cooperate to determine the appropriate learning objects to be presented to the student. Each agent is complex with it's own internal architecture, e.g., the learning object agent contains layers for managing communication, learning paths, and learning objects.

In the *multiple homogeneous agents* approach, multiple agents all have the same role and operate in parallel. These are the simulated and physical demonstrating agent systems described in Section 4.3. Typically, these types of systems do not contain the components outlined in Figure 2b, such as a student model or teaching component. As discussed in the earlier section, demonstrating agents comprise or embody the entire learning system, which in the terms of Figure 2b, consists only of domain knowledge and a user interface. These systems do not possess any reasoning capabilities about the learner. An example is NetLogo, where each agent is actually programmed by the student. The agents themselves are typically quite primitive, with reactive behaviors that respond to other agents and their environment but do not perform any sophisticated reasoning. However, complex system-wide behaviors can emerge through these interactions.

Figure 3 illustrates the agent-based architectural patterns, alongside the component model from Figure 2b. The single agent pattern is achieved by implementing one agent that handles all the capabilities inside the shaded box. The multiple heterogenous agents pattern is achieved by implementing multiple agents, frequently one for each of the components within the shaded box.

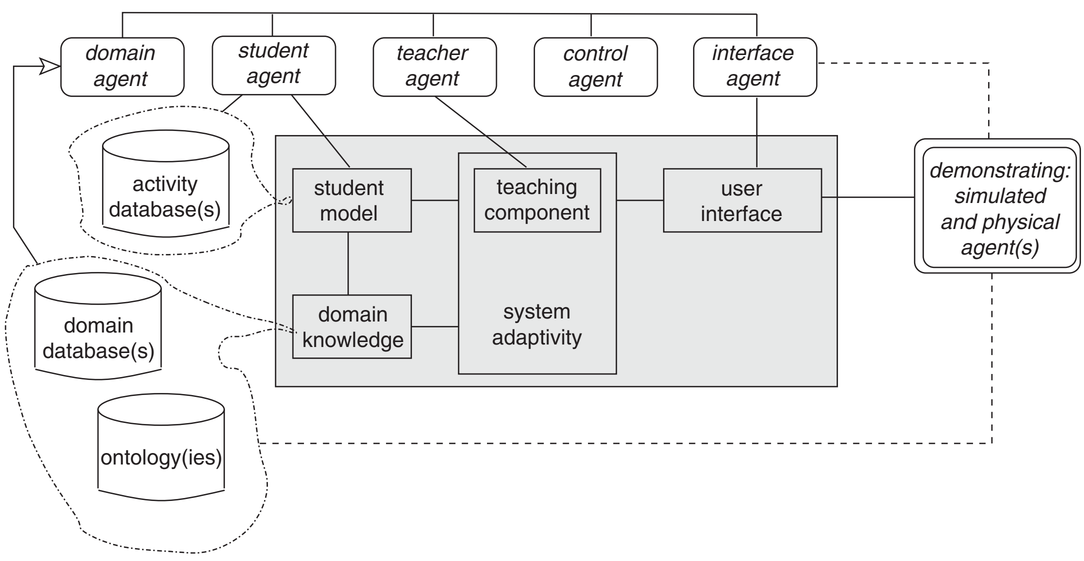

Figure 3 MAS architectural pattern

Table 1 Comparison of systems discussed

	Agent role	Architectural pattern
Pedagogical		
ABITS (Mowlds *et al.*, 2006)	Observational	Multiheterogeneous
Intelligent Scaffolding Agent (Blake *et al.*, 2007)	Observational	Single
Carmen's Bright IDEAS (Marsella *et al.*, 2000)	Participatory	Single
DAL (Sun *et al.*, 2005)	Observational	Multiheterogeneous
e-Advisor (Lin *et al.*, 2006, 2007)	Participatory	Multiheterogeneous
I-MINDS (Khandaker & Soh, 2007)	Mixed	Multiheterogeneous
MAIDE (Lucas *et al.*, 2005)	Mixed	Multiheterogeneous
Online Mediator Education System (Tanaka *et al.*, 2006)	Mixed	Single
STEVE (Rickel & Johnson, 1998)	Participatory	Single
TactLang (Johnson *et al.*, 2004)	Mixed	Single
VirSchool (Fassbender and Richards, 2006)	Participatory	Multiheterogeneous
Peer-learning		
Airport World (Richards *et al.*, 2007)	Participatory	Multiheterogeneous
Autism Project (Sehaba & Estraillier, 2005)	Mixed	Multiheterogeneous
Crystal Island (Mott *et al.*, 2006)	Participatory	Multiheterogeneous
PAL (Kim, 2005)	Participatory	Single
RAP (Lazzari *et al.*, 2005)	Participatory	Multiheterogeneous
SimEd (Spoelstra & Sklar, 2007)	Observational	Multihomogeneous
Demonstrating		
I-Shadows (Brisson *et al.*, 2007)	Participatory	Multihomogeneous
MIRA (Holz *et al.*, 2006)	Participatory	Multiheterogeneous
NetLogo (Wilensky, 2002; Blikstein & Wilensky, 2005; Sengupta & Wilensky, 2005)	Participatory	Multihomogeneous
REAL (Bai & Black, 2006; Bai *et al.*, 2007)	Participatory	Multihomogeneous
RoboXAP (Chu *et al.*, 2005)	Participatory	Single

Refer to Section 4 for descriptions of systems listed.

These are shown on the top row and include separate agents to interface with the domain database(s), to model the student, to implement a particular teaching style, and to communicate with the human user (i.e., student). Sometimes there is a central 'control agent' that oversees smooth operation of the entire system. On the right, there is one box with a double border containing the demonstrating agents, and we note that there are hybrid systems, particularly those employing simulated agents, that combine demonstrating agents with the other components or types of agents. However, there is no work that we are aware of which implements a physical agent in the same system as the other types of components (or agents), or one in which a pedagogical agent helps the user learn how to control a simulated agent.

Another way of comparing systems is by examining the role that agents play in interacting with humans. There are essentially two categories: *observational* and *participatory*. Systems with observational agents are likely to include, as represented in Figure 3, one or more knowledge bases, databases, and/or ontologies. These may be external to or embedded within an agent. These agents tend to be very complex and contain sophisticated reasoning capabilities. Systems with participatory agents include demonstrating agents as well as pedagogical agents that interact directly with the learner. Some systems are mixed, containing agents of each type.

Table 1 compares the systems highlighted here, particularly those discussed in Section 4. The systems are grouped based on type (pedagogical, peer-learning, or demonstrating), and two other axes are presented: agent role and architectural pattern. The only clear distinction is with the demonstrating systems: all are participatory in terms of agent role, and the multihomogeneous architectural pattern appears only in this type. Otherwise, there is a healthy mix of techniques in the pedagogical and peer-learning systems.

6 Testing and evaluation

One of the more prominent issues that separate human learning systems from other agent-based and interactive systems development is the aspect of testing and evaluation. Simply installing technology in educational settings does not ensure better learning outcomes. It is expected that not only the software learning environment be fully debugged and tested, but also, particularly among education researchers, the system must be evaluated with respect to its effectiveness as a learning environment. This section provides a brief description of evaluation in the broader context of interactive learning systems, with emphasis on aspects of interest to developers of agent-based systems for human learning.

While there is no fixed standard for evaluating the effectiveness of interactive learning systems, there are two generally accepted categories of assessment (Littman & Soloway, 1988; Mark & Greer, 1993):

- *formative assessment* tests the design and behavior of a system in-progress, generally performed by computer scientists, system designers, and builders; and
- *summative assessment* evaluates the effectiveness of a completed system, generally performed by educators and/or psychologists.

Researchers begin by identifying what is being evaluated. Design and performance aspects need to be examined differently. The nature of the testing will vary depending on whether the goal is to assess the theoretical basis underlying the system or the software components themselves.

Within formative assessment, each of a learning system's components (identified in Figure 2b) can be evaluated individually. Domain knowledge should be checked for accuracy and coverage. The teaching component can be evaluated for the range of instructional method(s) offered, its level of adaptability and the degree to which its instruction is based on proven educational and psychological methods. The user interface can be examined by comparing multiple user interfaces for the same underlying engine and looking, in particular, for improvement in student learning. System adaptivity can be compared using interactions at different skill levels. The control component can be evaluated using various system performance measures, such as speed. Finally, and probably the most important, improvement in student knowledge (i.e., learning) can be measured using the same criteria in a computer-based environment that are employed within standard educational and/or psychological testing. These include: (1) validity—does the test show evidence that it measures what it says it measures? (2) reliability—are multiple results for the same subject consistent? (3) objectivity—is the test administered and scored the same way for every participant? (4) standardization—can results be translated into a meaningful representation of student performance?

Pedagogical drama is a specialized case of the more general area known as *interactive drama*, which includes immersive environments for entertainment as well as education. Recent work has examined *drama management (DM)* within interactive drama systems (Roberts & Isbell, 2007) and included discussion about the application of DM research to educational systems. (Roberts & Isbell, 2007) list a number of measures against which a DM system could be evaluated: speed (of system operation), coordination (among players and non-player characters in a system), replayability, authorial control, player autonomy, ease of authoring, adaptability, soundness, invisibily, and measurability (of user's satisfaction with the system). All of these could be adapted for use in evaluating the immersive systems described here, where the learner takes on the role of a player, the teacher takes on the role of an author, and non-player characters are either learning peers, advisers or tutors, or background elements of the system.

The techniques for performing assessments vary depending on which component is being evaluated, the phase in the system development cycle in which the evaluation is being performed and who is performing the evaluation. Similar to typical HCI lifecycle models, such as the star life cycle (Preece *et al*., 1994) that involves evaluation after each step, evaluation of learning systems typically follows an iterative cycle. Beginning with system development and extending through to experimental research, steps may be revisited at any time during the formative phases of system development. Once summative assessment begins, in the experimental research phase, the system cannot change; otherwise, the summative results will be invalid. Pilot testing often occurs late during formative assessment, bridging the gap to summative assessment. There are three methods of pilot testing (Sklar, 2000): (1) one-to-one, which is performed early in the development cycle, with one student, instructor, trainer, or researcher providing feedback; (2) small-group, which is performed later in the development cycle, with a small group of students, instructors, or trainers providing feedback; and (3) field, which is performed near the end of development, emulating experimental conditions with teachers or trainers and students in a 'live' (i.e., classroom) setting.

Other techniques are more pertinent during summative assessment. In criterion-based evaluation, a general list of guidelines is developed and systems are evaluated based on their adherence to these guidelines, for example, program construction, behavior, and characteristics. While developing relevant criteria is not an easy task, this method may prove useful in formative assessment and in comparing different systems. With expert knowledge and behavior assessment, system performance is compared with that of a human expert performing the same task. Software systems may be subjected to a standard certification process, through careful examination by qualified human experts. In sensitivity analysis, the responsiveness of a system is tested on a variety of different user behaviors. This may be particularly useful for evaluating system adaptivity. After system development and pilot testing are complete, experimental research can begin. The conditions should be the same as those during the field testing phase.

Two mechanisms for collecting evaluation data are common:

- *quantitative*, in which numerical data is analysed, frequently by comparing scores on pre- and post-tests and surveys, to measure changes in student performance and attitudes; and
- *qualitative*, in which interviews and surveys are conducted and observations are made.

Mixed methods research (Johnson & Onwuegbuzie, 2004) combines the two, but traditionally, at least in the education arena, researchers tend to adhere to the methods of one category or the other. Quantitative methods rely on standards testing styles, with multiple choice questions and Likert scale surveys. System logs are also examined. Qualitative, or 'open', methods encompass data taken in both written and oral forms, as part of interviews, questionnaires and open surveys containing short-answer questions (rather than multiple choice). Transcripts are 'coded' and analysed based on measures such as frequency of broad term usage, often borrowing techniques from natural language processing in order to compute semantic similarity between answers.

Reviewing the literature describing agent-based systems issues, written from an agents research perspective (as opposed to an education research perspective), the primary type of evaluation is formative assessment, particularly testing the accuracy and functionality of system design, and early pilot testing. Within the sampling of literature referenced in this article, a wide range of test environments have been used, including primary, secondary and undergraduate classrooms, research laboratories, industrial workplaces, military bases, and distance learning (or 'e-learning') settings. As well, there are broad differences in the maturity of systems presented. Some have completed only the design phase while others have been fully implemented. Many have undergone some aspects of formative testing, including architecture and system design reviews, and user interface studies. Few have reached the summative testing phases, but some have conducted focus groups and pilot studies.

Despite the growing interest in the inclusion of intelligent agent technology in e-learning environments (e.g., Conole, 2002; Logan *et al.*, 2002; Songa *et al.*, 2004), Mahmood and Ferneley (2006) believe, with respect to animated agents for e-learning, that 'empirical investigations of their use in online education are limited' and that 'there is now a sense of urgency in identifying how to incorporate pedagogical agent technology appropriately in e-learning environments' (pp. 153–154). In response to this need, (Mahmood & Ferneley, 2006) have conducted an empirical exploratory study to investigate appropriate use of pedagogical agents and propose a general framework which evaluates: animated agent roles, dialogue context, visual context, pedagogical context, animated agent service quality and usefulness, presence context, and profile context.

Many systems that are designed do not get past the prototype phase for various reasons. First, gaining access to human subjects, particularly minors, may be difficult. Hurdles can include: availability of a representative population, costs for setup, recruitment, human ethics requirements, and access to a control group. One of the more serious issues is that learning systems take so long to develop, by the time they are operational, the customer does not need or want them any more. This is frequently due to the large costs (in time and money) associated with building a teaching component into a system, which adds to the typically unwieldy costs of developing software on time and within budget (Brooks, 1975). Nonetheless, just as one would expect user evaluation to be found in research publications concerning user interfaces, there is a higher expectation in the work reported on agent-based human learning systems that an evaluation with real users will be conducted, will be well designed and analyzed carefully.

7 Open issues and future directions

The types of research, application development, and studies being performed using agent technology for human learning are varied. In some cases, the central theme of a project is to explore and extend current agent technologies and capabilities. In other cases, agent theory is not being developed but rather is being applied and the research focuses on the application of theories.

Differences in learners' ages and genders must be considered when designing agents that will interact as pedagogical tutors or peer learners. While most learning systems using games are focused on making the learning more palatable for children, the motivation of researchers building systems for adults (e.g., Richards *et al.*, 2005) is quite different. Kearsley (2004) emphasizes that adult learners tend to focus more on the process of learning, in which strategies such as reflection, role-playing, simulation, and case studies are most useful, with instructors acting as facilitators. Using virtual environment technologies in conjunction with agent components allows production of less expensive, more flexible, and more accessible systems that offer increased control of the environment together with increased relevance to the real world.

Many studies have highlighted differences in the way females and males approach, interact with, and think about technology. (Inkpen, 1994) found gender differences in the way children approach game environments. Girls tended to perform better when another person was also playing on the same machine, but boys performed better when the other player was on a different machine. Girls were found to have less physical contact with their human partner or the mouse compared to boys. Brunner *et al.* (1998) showed that females tend to view technology as a tool used to facilitate human interaction, whereas males tend to view technology as an object that can be used to extend their abilities and/or power. These gender-based attitudinal differences will affect a student's experience with a learning environment; as well, these issues generalize to any interactive agent-based system.

A more philosophical question addresses the intersection of agent-based technologies and interactive learning systems. What does each do for the other? Agent-based systems, since the infant stages of the field (Maes, 1994b; Wooldridge & Jennings, 1995), have been highly focused on finding effective, modular means to interact intelligently with human users in a personalized, but not annoying way. The notion of an automated assistant is quite natural in a learning or tutoring environment. The difficulty comes in building generalizable systems because effective tutoring, by both humans and agents, involves deep understanding of the knowledge domain being taught and the student being guided.

Knowledge engineering, identifying and fixing common bugs in students' learning paths and constructing tutoring agents that reflect the experience of a master teacher is a daunting endeavor, shadowed by the desire to avoid repeating the mistakes of expert systems. Agents have been shown to learn to adapt in dynamic environments, such as robotic soccer (Stone *et al.*, 2005) and electronic markets (Walia *et al.*, 2003). If we view the human learner as an agent's changing environment, perhaps tomorrow's solutions will include agents that can learn to teach. An even greater challenge, which is the shared vision of Sklar (2000) and Richards *et al.* (2006), is an agent system that progresses with the learner, advancing its knowledge in growing domains and assisting the learner also to do so. One approach uses an incremental rule and case-based knowledge acquisition technique known as *ripple down rules*, the goal is to allow the trainer to incrementally add new objects, scenarios, knowledge (rules), and even agents to perform new tasks, as they interact with the system (Richards *et al.*, 2006). Another approach employs *evolutionary computation* for producing agents that adapt dynamically as the human learner progresses (Sklar, 2000).

MAS architectures demonstrate how a society of agents can work together to achieve a goal, much as a team or group of humans do. However, as with much AI research, the focus in MAS research tends to be on understanding human social behaviors in order to mimic this in computer systems (e.g. Jars *et al.*, 2004). However, we believe that a key area for future human learning systems is the use of MAS in pedagogical environments, with an emphasis on the social interactions between humans and groups of agents, as well as among agents themselves. One such area ripe for exploring the social side of agent technology is what has become known as social software, which includes Wikis, WebLogs, and online Communities of Practice.

Social software has emerged via the web, driven by and concurrently sustaining global knowledge-sharing communities. Agent systems are likely to play a major role in this area, similar to their envisaged role in turning the current web into the next generation, together with other semantic web technologies (Berners-Lee & Fischetti, 1999). As with most of the current web content, the knowledge in social systems is captured for human consumption only. Social software, however, not only lacks semantics but also, in most cases, lacks structure and syntax making automated reasoning by software agents difficult. Pacuit (2007) explores the use of agents in social software environments, focusing on traditional MAS, logic, and game theory. This approach seems based on his definition that 'social software is an interdisciplinary research program that combines mathematical tools and techniques from game theory and computer science in order to analyze social procedures' (p. 2). This definition contrasts with the view purported in the educational or knowledge management literature where the emphasis is on modeling social networks (Scott, 1991) to identify communication patterns such as bottlenecks or cliques and the management of mentoring and team building programs. To date, there appears to be little work investigating the role of agents in social software to facilitate human learning.

In keeping with the knowledge management view, Anderson (2005) talks about distance learning being the potential 'killer application' for social software. In particular, Anderson focuses on educational social software and gives a number of definitions from the literature such as 'tools which support communication using the five devices of identity, presence, relationships, conversations, and groups' (p. 4). In the future, we can expect agent-based systems that contain sophisticated agents embodying these five devices or multiple agents that work together to provide such a learning tool.

Further outstanding issues have been identified by Mahmood and Ferneley (2006). These include: (1) potentially conflicting roles of various animated agents, which becomes more of a problem as more agents join the dialogue; (2) degradation in quality of service where multiple teachers/experts may be needed to provide support due to inconsistencies, in format, style, and content, and the effort required to identify and reconcile such differencies; (3) assurance of the integrity of (animated) agents; and (4) development of personalized presentation styles based on users' profiles that include learning styles, career background, and interests.

In summary, this article has attempted to characterize the types of agent-based systems that are being developed to support a wide range of human learners from young children in formal classroom or informal after-school settings to lifelong (adult) learners in workplaces. Many different approaches bring a spectrum of agent theories into practice by their application to this challenging domain. Common interaction methods and system architectural patterns are highlighted. The particular challenges for building applications in the area of agent-based systems for human learning encompass not only those faced by developers of any type of interactive agent-based system, but also unique aspects tailored to learners—users who are advancing as they interact with a system, users who may have academic, social or motivational deficiencies, and users who may have limited technical sophistication. In this domain, system testing requirements are more rigorous than those for general interactive systems—not only is formative evaluation performed, to test whether or not the software works, but also summative evaluation, to test whether (and how well) the software is helping users learn. By pointing out these special needs of agent-based systems designed for human learners and highlighting open issues, it is hoped that this article helps to lay the groundwork for establishing a common vocabulary and shared dialogue for developers within this exciting and promising application area.

Acknowledgments

This work was partially supported by NSF #ITR-02-19347/05-52294 and ARC DP0558852. Thanks to anonymous reviewers who have provided feedback on previous versions of this paper.

References

Adobbati, R., Marshall, A. N., Scholer, A., Tejada, S., Kaminka, G. A., Schaffer, S. & Sollitto, C. 2001. Gamebots: a 3d virtual world test-bed for multi-agent research. In *Proceedings of the Second International Workshop on Infrastructure for Agents, MAS, and Scalable MAS*, Wagner, T. & Rana, O. F. (eds). Montreal, Canada.

Albus, J. S., McCain, H. & Lumia, R. 1987. *Nasa/nbs Standard Reference Model for Telerobot Control System Architecture (nasrem)*. Technical report, Robot Systems Division, National Bureau of Standards, Gaithersburg, MD, USA.

Aldrich, C. 2003. *Simulations and the Future of Learning: An Innovative (and Perhaps Revolutionary) Approach to E-Learning*. Pfeiffer.

Anderson, J. R. 1982. Acquisition of cognitive skill. *Psychology Review* **89**(4), 369–406.

Anderson, T. 2005. Distance learning—social software's killer app. In *Proceedings of 17th Biennial Conference of the Open and Distance Learning Association of Australia*, ODLAA: Adelaida.

Arkin, R. 1995. Reactive robotic systems. In *Handbook of Brain Theory and Neural Networks*. MIT Press.

Arkin, R. 1998. *Behavior-based Robotics*. MIT Press.

Arkin, R. C. 1989. Towards the unification of navigational planning and reactive control. In *Proceedings of the AAAI Spring Symposium on Robot Navigation*, AAAI Press, 1–5.

Azhar, M. Q., Goldman, R. & Sklar, E. 2006. An agent-oriented behavior-based interface for educational robotics. In *Proceedings of the Agent-based Systems for Human Learning (ABSHL) Workshop at AAMAS-2006*, 41–46.

Bai, X. & Black, J. 2006. Real: an agent-based learning environment. In *Proceedings of the Agent-based Systems for Human Learning (ABSHL) Workshop at AAMAS-2006*, 13–17.

Bai, X., Black, J. B. & Vitale, J. 2007. Learning with the assistance of a reflective agent. In *Proceedings of the Agent-based Systems for Human Learning Workshop (ABSHL) at Autonomous Agents and Multiagent Systems (AAMAS)*, 22–29.

Balabanovic, M. 1998. *Learning to Surf: Multi-agent Systems for Adaptive Web Page Recomendation*. PhD thesis, Stanford University.

Barles, J., Dras, M., Kavakli, M., Richards, D. & Tychsen, A. 2005. An overview of training simulation research and systems. In *Proceedings of the Agent-based Systems for Human Learning (ABSHL) Workshop at AAMAS-2005*, 3–10.

Bartles, R. 2003. *Designing Virtual Worlds*. New Riders.

Beck, J., Stern, M. & Haugsjaa, E. 1996. Applications of AI in education. *Crossroads: Special Issue on Artificial Intelligence* **3**(1), 11–15.

Beck, J. E. 1998. Learning to teach with a reinforcement learning agent. In *Proceedings of Fifteenth National Conference on Artificial Intelligence*, AAAI Press, Menlo Park, California.

Berners-Lee, T. & Fischetti, M. 1999. *Weaving the Web*. Harper.

Blake, M. B., Butcher-Green, J. D. & Doswell, J. T. 2007. Agent-mediated customized training for human learning performance enhancement. In *Proceedings of the Agent-based Systems for Human Learning Workshop (ABSHL) at Autonomous Agents and Multiagent Systems (AAMAS)*, 80–85.

Blank, D. S., Kumar, D., Meeden, L. & Yanco, H. 2005. Pyro: a python-based versatile programming environment for teaching robotics. *Journal on Educational Resources in Computing (JERIC), Special Issue on Robotics in Undergraduate Education, Part I* **3**(4), 1–15.

Blikstein, P. & Wilensky, U. 2005. Less is more: agent-based simulation as a powerful learning tool in materials science. In *Proceedings of the Agent-based Systems for Human Learning (ABSHL) Workshop at AAMAS-2005*, 11–18.

Bratman, M. E. 1987. *Intentions, Plans and Practical Reason*. Harvard University Press.
Brisson, A., Dias, J. & Paiva, A. 2007. From Chinese Shadows to Interactive Shadows: building a storytelling application with autonomous shadows. In *Proceedings of the Agent-based Systems for Human Learning Workshop (ABSHL) at Autonomous Agents and Multiagent Systems (AAMAS)*, 15–21.
Brooks, F. P. 1975. *The Mythical Man-Month: Essays on Software Engineering*. Addison-Wesley.
Brooks, R. 1984. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation* **2**(1), 14–23.
Brown, J. S. & Burton, R. B. 1978. Diagnostic models for procedural bugs in basic mathematical skills. *Cognitive Science* **2**(2), 155–192.
Brunner, C., Bennett, D. & Honey, M. 1998. Girls' games and technological desire. In *From Barbie to Mortal Kombat: Gender and Computer Games*, Cassell, J. & Jenkins, H. (eds). MIT Press, 72–88.
Chu, K.-H., Goldman, R. & Sklar, E. 2005. Roboxap: an agent-based educational robotics simulator. In *Proceedings of the Agent-based Systems for Human Learning (ABSHL) Workshop at AAMAS-2005*, 19–26.
Clancey, W. J. 1986. *Intelligent Tutoring Systems: A Tutorial Survey*. Technical Report STAN-CS-87-1174, Stanford University.
Conole, G. 2002. The evolving landscape of learning technology. *Association for Learning Technology Journal* **10**, 4–18.
Cypher, A. 1991. Eager: programming repetitive tasks by example. In *Proceedings of Computer-Human Interaction (CHI)*, New Orleans, April 28–May 2, ACM, New York, 33–39.
Epstein, J. M. & Axtell, R. 1996. *Growing Artificial Societies: Social Science from the Bottom Up*. Technical report, The Brookings Institution, Washington, DC, USA.
Erwin, B., Cyr, M. & Rogers, C. B. 2000. Lego engineer and robolab: teaching engineering with labview from kindergarten to graduate school. *International Journal of Engineering Education* **16**(3), 181–192.
Fassbender, E. & Richards, D. 2006. Agent-driven knowledge exchange in a 3d learning environment. In *Proceedings of the Agent-based Systems for Human Learning (ABSHL) Workshop at AAMAS-2006*, 59–64.
Feurzeig, W., Papert, S., Bloom, M., Grant, R. & Solomon, C. 1970. Programming-languages as a conceptual framework for teaching mathematics. *ACM SIGCUE Outlook* **4**(2), 13–17.
Foner, L. 1997. Yenta: A multi-agent referral based matchmaking system. In *Proceedings of the First International Conference on Autonomous Agents*. ACM.
Galarneau, L. 2004. The elearning edge: leveraging interactive technologies in the design of engaging, effective learning experiences. In *Proceedings of e-Fest 2004*. Udas, K. (ed.). Wellington, New Zealand, ITP, 1–10.
Gamma, E., Helm, R., Johnson, R. & Vlissides, J. 1995. *Design Patterns: Elements of Reusable Object Oriented Software*. Addison-Wesley.
Goldberg, D., Nichols, D., Oki, B. M. & Terry, D. 1992. Using collaborative filtering to weave and information tapestry. *Communications of the ACM* **35**(12), 61–70.
Goldman, R. 2005. *From robolab to aibo: Capturing Agent Behavior*. Master's thesis, Computer Science Department, Columbia University, New York, NY, USA.
Goldman, R., Eguchi, A. & Sklar, E. 2004. Using educational robotics to engage inner-city students with technology. In *Proceedings of the Sixth International Conference of the Learning Sciences (ICLS)*.
Hartshorn, R. & Boren, S. 1990. Experiential learning of mathematics: using manipulatives. *ERIC Clearinghouse on Rural Education and Small Schools* **ED321967**, 1–7.
Hayden, S. C., Carrick, C. & Yang, Q. 1999. Architectural design patterns for multi-agent coordination. In *Proceedings of the International Conference on Agent Systems (Agents'99)*, ACM Press.
Holz, T., Dragone, M., O'Hare, G., Martin, A. & Duffy, B. 2006. Mixed reality agents as museum guides. In *Proceedings of the Agent-based Systems for Human Learning (ABSHL) Workshop at AAMAS-2006*, 33–40.
ID Software 2009. `http://www.idsoftware.com/business/history/`
Inkpen, K. 1994. Gender differences in an electronic games environment. In *Proceedings of ED-MEDIA 94 World Conference on Educational Multimedia and Hypermedia*, 25–30 June 1994, Ottman, T. & Tomek, I. (eds). AACE.
Jarrold, W. L. 2007. Treating Autism with the help of artificial intelligence: a value proposition. In *Proceedings of the Agent-based Systems for Human Learning Workshop (ABSHL) at Autonomous Agents and Multiagent Systems (AAMAS)*, 30–37.
Jars, I., Kabachi, N. & Lamure, M. 2004. Social learning in multi-agents system: an additional step toward intelligence, lass. In *Systems, Man and Cybernetics, 2004 IEEE International Conference*, **6**, 5542–5547. Claude Bernard Lyon I University.
Johnson, R. B. & Onwuegbuzie, A. J. 2004. Mixed methods research: a research paradigm whose time has come. *Educational Researcher* **33**(7), 5.
Johnson, W. L. 1995. Pedagogical agents for virtual learning environments. In *Proceedings of the International Conference on Computers in Education*, Singapore, 41–48.

Johnson, W. L. 2001. Pedagogical agent research at carte. *AI Magazine* **22**(4).

Johnson, W. L., Beal, C., Fowles-Winkler, A., Narayanan, S., Papachristou, D., Marsella, S. & Vilhj, H. 2004. Training system: an interim report. In *Proceedings of the International Conference on Intelligent Tutoring Sys*, Springer Verlag: Berlin, 336–345.

Kearsley, G. 2004. *Explorations in Learning and Instruction: the Theory into Practice Database*. `http://tip.psychology.org/index.html`

Khandaker, N. & Soh, L.-K. 2007. Formation and scaffolding human coalitions in I-MINDS – a computer-supportive collaborative learning environment. In *Proceedings of the Agent-based Systems for Human Learning Workshop (ABSHL) at Autonomous Agents and Multiagent Systems (AAMAS)*, 69–79.

Kim, Y. 2005. Agent learning companions: learners expectations of the desirable characteristics. In *Proceedings of the Agent-based Systems for Human Learning (ABSHL) Workshop at AAMAS-2005*, 27–34.

Klassner, F. & Anderson, S. 2003. Lego mindstorms: not just for k-12 anymore. *IEEE Robotics and Automation* **10**(2), 12–18.

Kolodner, J. L. 1983. Maintaining organization in a dynamic long-term memory. *Cognitive Science* **7**(4), 243–280.

Kuokka, D. & Harada, L. 1997. Matchmaking for information agents. In *Readings in Agents*. Morgan Kaufman.

Lang, K. 1995. Newsweeder: Learning to filter news. In *Proceedings of the Twelth International Conference on Machine Learning (ICML)*, Morgan Kaufmann Publishers Inc., 331–339.

Lashkari, Y., Metral, M. & Maes, P. 1994. Collaborative interface agents. In *Proceedings of the Twelth National Conference on Artificial Intelligence (AAAI)*, Menlo Park, CA, 444–449.

Lave, J. & Wenger, E. 1991. *Situated Learning: Legitimate Peripheral Participation*. Cambridge University Press.

Lazzari, L., Mari, M., Negri, A. & Poggi, A. 2005. Support remote software development in an open distributed community. In *Proceedings of the Agent-based Systems for Human Learning (ABSHL) Workshop at AAMAS-2005*, 68–73.

Lieberman, H. 1995. Letizia: an agent that assists web browsing. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI)*, Mellish, C.S. (ed.). San mateo, CA, Kaufmann Publishers, 924–929.

Lin, F., Leung, S., Wen, D., Zhang, F., Kinshuk, & McGreal, R. 2007. e-Advisor: a Multi-agent system for academic advising. In *Proceedings of the Agent-based Systems for Human Learning Workshop (ABSHL) at Autonomous Agents and Multiagent Systems (AAMAS)*, 86–94.

Lin, F., Zhang, F. & Wen, D. 2006. Developing mas for academic advising. In *Proceedings of the Agent-based Systems for Human Learning (ABSHL) Workshop at AAMAS-2006*, 53–58.

Littman, D. & Soloway, E. 1988. Evaluating itss: the cognitive science perspective. In *Foundations of Intelligent Tutoring Systems*. Lawrence Erlbaum Associates.

Logan, B., Fraser, M., Fielding, D., Benford, S., Greenhalgh, C. & Herrero, P. 2002. Keeping in touch: agents reporting from collaborative virtual environments. In *Artificial intelligence and interactive entertainment, papers presented at the American Association for Artificial Intelligence Symposium*, Technical report SS-02-01, AAAI Press, Menlo Park, CA, USA.

Lucas, J. P., Wilges, B. & Silveira, R. A. 2005. Inserting animated pedagogical agents inside distributed learning environments by means of fipa specifications. In *Proceedings of the Agent-based Systems for Human Learning (ABSHL) Workshop at AAMAS-2005*, 74–77.

Maes, P. 1994a. Modeling adaptive autonomous agents. *Artificial Life Journal* **1**(1–2), 135–162.

Maes, P. 1994b. Modeling adaptive autonomous agents. *Artificial Life: An Overview*.

Mahmood, A. K. & Ferneley, E. 2006. The use of animated agents in e-learning environments: an exploratory interpretive case study alt-j. *Research in Learning Technology* **14**(2), 153–168.

Malone, T. 1981a. Toward a theory of intrinsically motivating instruction. *Cognitive Science* **5**(4), 333–369.

Malone, T. 1981b. What makes computer games fun? *Byte Magazine* **6**, 258–277.

Mark, M. A. & Greer, J. E. 1993. Evaluation methodologies for intelligent tutoring systems. *Journal of Artificial Intelligence and Education* **4**, 1–30.

Marsella, S., Johnson, W. L. & LaBore, K. 2000. Interactive pedagogical drama. In *Proceedings of the Fourth International Conference on Autonomous Agents*. ACM Press: New York.

Martin, F. 1994. *Circuits to Control: Learning Engineering by Designing LEGO Robots*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA.

Mataric, M. J. 1998. Behavior-based robotics as a tool for synthesis of artificial behavior and analysis of natural behavior. *Trends in Cognitive Science* **2**(3), 82–87.

McCalla, G. I. & Greer, J. E. 1994. Granularity-based reasoning and belief revision in student models. In *Student Models: The Key to Individualized Educational Systems*, 39–62. Springer Verlag.

Mott, B., McQuiggan, S. & Lee, S. 2006. Narrative-centered environment for guided exploratory learning. In *Proceedings of the Agent-based Systems for Human Learning (ABSHL) Workshop at AAMAS-2006*, 26–32.

Mowlds, F., Roche, B.-J. & Mangina, E. 2006. Use of agents in e-learning to enrich students learning experiences. In *Proceedings of the Agent-based Systems for Human Learning (ABSHL) Workshop at AAMAS-2006*, 5–12.

Nilsson, N. J. 1969. A mobile automaton. In *Proceedings of the 1st International Joint Conference on Artificial Intelligence (IJCAI)*, Walker, D. E. & Norton, L. M. (eds), May 1969, Washington, DC, William Kaufmann.

Nilsson, N. J. 1984. *Shakey the Robot*. Technical report, Artificial Intelligence Center, SRI International, Menlo Park, CA, USA.

North, M. J., Collier, N. T. & Vos, J. R. 2006. Experiences creating three implementations of the repast agent modeling toolkit. *ACM Transactions on Modeling and Computer Simulation* **16**(1), 1–25.

Pacuit, E. 2007. Some comments on history based structures. *The Journal of Applied Logic*, 613–624.

Papert, S. 1991. Situating constructionism. In *Constructionism*. Ablex Publishing.

Preece, J., Rogers, Y., Sharp, H., Benyon, D., Holland, S. & Carey, T. 1994. *Human-Computer Interaction*. Addison-Wesley.

Pynadath, D. V. & Marsella, S. C. 2005. Psychsim: modeling theory of mind with decision-theoretic agents. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI)*, July 30–August 5, 2005, Edinburgh, Scotland, UK, 1181–1186.

Repenning, A. & Citrin, W. 1993. Agentsheets: applying grid-based spatial reasoning to human-computer interaction. In *Proceedings of the IEEE Workshop on Visual Languages*, August 24–27, 1993, Bergen, Norway.

Resnick, M. 1997. *Turtles, Termites and Traffic Jams: Explorations in Massively Parallel Microworlds*. The MIT Press.

Richards, D., Barles, J., Szilas, N., Kavakli, M. & Dras, M. 2006. Human and software agents learning together. In *Proceedings of the Agent-based Systems for Human Learning (ABSHL) Workshop at AAMAS-2006*, 18–25.

Richards, D., Kavakli, M. & Dras, M. 2005. Training for high risk situations. In *Proceedings of the Agent-based Systems for Human Learning (ABSHL) Workshop at AAMAS-2005*, 35–41.

Richards, D., Kavakli, M. & Dras, M. 2007. Impacts of visualisation, interaction and immersion on learning using an agent-based training simulation. In *Proceedings of the Agent-based Systems for Human Learning Workshop (ABSHL) at Autonomous Agents and Multiagent Systems (AAMAS)*, 57–68.

Rickel, J. & Johnson, W. L. 1998. STEVE: a pedagogical agent for virtual reality. In *Proceedings of the Second International Conference on Autonomous Agents*, St. Paul, Minnepolis, USA, May 9–13, Sycara, K. P., Wooldridge, M. (eds). New York, ACM Press.

Roberts, D. L. & Isbell, C. L. 2007. Desiderata for managers of interactive experiences: a survey of recent advances in drama management. In *Proceedings of the Agent-based Systems for Human Learning Workshop (ABSHL) at Autonomous Agents and Multiagent Systems (AAMAS)*, 5–14.

Russell, S. & Norvig, P. 2002. *Artificial Intelligence: A Modern Approach*, 2nd edn, Prentice Hall.

Schank, R. C. 1981. Failure-driven memory. *Cognition and Brain Theory* **4**(1), 41–60.

Scott, J. 1991. *Social Network Analysis: a handbook*. SAGE Publications.

Sehaba, K. & Estraillier, P. 2005. A multi-agent system for rehabilitation of children with autism. In *Proceedings of the Agent-based Systems for Human Learning (ABSHL) Workshop at AAMAS-2005*, 84–89.

Sengupta, P. & Wilensky, U. 2005. Niels: an emergent multi-agent based modeling environment for learning physics. In *Proceedings of the Agent-based Systems for Human Learning (ABSHL) Workshop at AAMAS-2005*, 42–47.

Shaw, M. 1996. Some patterns for software architectures. In *Proceedings of the Second Workshop on Pattern Languages for Programming*. Addison Wesley.

Shiri, M., Aïmeur, E. & Frasson, C. 1998. Student modelling by cased based reasoning. In *Fourth International Conference on Intelligent Tutoring Systems (ITS-98)*, San Antonio, Texas, USA, August 16–19, Goettl, B. P., Half, H. M., Redfield, C. L., Shute, V. J. (eds). Lecture Notes in Computer Science **1452**, 394–403. Springer.

Si, M., Marsella, S. C. & Pynadath, D. V. 2005a. Thespian: using multi-agent fitting to craft interactive drama. In *Proceedings of the 4th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, July 25–29, 2005, Utrecht, The Netherlands. Dignam, F., Dignum, V., Koenig, S., Kraus, S., Singh, M. P. & Wooldridge, M. (eds). ACM.

Si, M., Marsella, S. C. & Pynadath, D. V. 2005b. Thespian: an architecture for interactive pedagogical drama. In *Proceedings of the 2005 International Conference on Artificial Intelligence in Education (AIED)*: Supporting Learning through Intelligent and Socially Informed Technology. Looi, C., MeCalla, G., Bredeweg, B. & Breuker, J. (eds). Frontiers in Artificial Intelligence and Applications, **125**, 592–602, Amsterdam, The Netherlands, IOS Press.

Sklar, E. 2000. *CEL: A Framework for Enabling an Internet Learning Community*. PhD thesis, Department of Computer Science, Brandeis University, Waltham, MA, USA.

Sklar, E. 2003. Agents for education: when too much intelligence is a bad thing. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, 1118–1119, ACM Press.

Sklar, E. 2007. Software review: Netlogo, a multiagent simulation environment. *Journal of Artificial Life* **13**(2), 303–311.

Sklar, E., Davies, M. & Co, M. S. T. 2004. SimEd: simulating education as a multiAgent system. In *Proceedings of the Third International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, 998–1005. ACM Press.

Sklar, E. & Parsons, S. 2002. RoboCupJunior: a vehicle for enhancing technical literacy. In *Proceedings of the AAAI-02 Mobile Robot Workshop*. AAAI Press.

Sklar, E., Parsons, S. & Stone, P. 2005. Using robocup in university-level computer science education. *Journal on Educational Resources in Computing (JERIC), Special Issue on robotics in undergraduate education, part I* **4**(2), 1–21.

Sklar, E. & Richards, D. 2006. The use of agents in human learning systems. In *Fifth International Conference on Autonomous Agents and Multi Agent Systems (AAMAS)*, Nakashima, H., Wellman, M. P., Weiss, G. & Stone, P. (eds). May 8–12, 2006, ACM, 767–774

Soloway, E. M., Woolf, B., Rubin, E. & Barth, P. 1981. Meno-ii: an intelligent tutoring system for novice programmers. In *Proceedings of the Seventh International Joint Conference on Artificial Intelligence (IJCAI)*, Hayes, P. J. (ed.). Vancouver, BC, Canada, Willam Kaufmann.

Sommerville, I. 1992. *Software Engineering*. Addison-Wesley.

Songa, K.-S., Hub, X., Olneyc, A. & Graesserb, A. C. 2004. A framework of synthesizing tutoring conversation capability with web-based distance education courseware. *Computers & Education* **42**, 375–388.

Spoelstra, M. & Sklar, E. 2007. Using simulation to model and understand group learning. In *Proceedings of the Agent-based Systems for Human Learning Workshop (ABSHL) at Autonomous Agents and Multiagent Systems (AAMAS)*, 38–47.

Stone, P., Kuhlmann, G., Taylor, M. E. & Liu, Y. 2005. Keepaway soccer: from machine learning testbed to benchmark. In *Proceedings of RoboCup-2005: Robot Soccer World Cup IX*. Springer Verlag.

Sun, S., Joy, M. & Griffiths, N. 2005. An agent-based approach to dynamic adaptive learning. In *Proceedings of the Agent-based Systems for Human Learning Workshop at AAMAS-2005*, 48–58.

Suydam, M. & Higgins, J. 1977. Activity-based learning in elementary school mathematics: Recommendations from research. *ERIC Clearinghouse for Science, Mathematics, and Environmental Education* **ED144840**, 203 pp.

Swartout, W., Hill, R., Gratch, J., Johnson, W. L., Kyriakakis, C., LaBore, C., Lindheim, R., Marsella, S., Miraglia, D., Moore, B., Morie, J., Rickel, J., Thiebaux, M., Tuch, L., Whitney, R. & Douglas, J. 2001. Toward the holodeck: integrating graphics, sound, character and story. In *Proceedings of the Fifth International Conference on Autonomous Agents*, May 28–June 1, 2001, Montreal, Canada. ACM.

Tanaka, T., Katagami, D. & Nitta, K. 2006. Advice agent for online mediator education. In *Proceedings of the Agent-based Systems for Human Learning (ABSHL) Workshop at AAMAS-2006*, 47–52.

van den Broek, E. 2005. Empathic agent technology (eat). In *Proceedings of the Agent-based Systems for Human Learning (ABSHL) Workshop at AAMAS-2005*, 59–67.

VanLehn, K. 1983. Human procedural skill acquisition: theory, model, and psychological validation. In *Proceedings of the National Conference on AI*, 420–423. Los Altos, CA, Morgan Kaufmann.

VanLehn, K., Niu, Z., Slier, S. & Gertner, A. 1998. Student modeling from conventional test data: a bayesian approach without priors. In *Proceedings of the 4th Intelligent Tutoring Systems Conference (ITS'98)*, 434–443. Springer-Verlag, Berlin, Heidelberg, GmbH & Co. K.

Vygotsky, L. S. 1978. *Mind in Society: The Development of Higher Psychological Processes*. Harvard University Press.

Wagner, S. P. 1999. Robotics and children: science achievement and problem solving. *Journal of Computing in Childhood Education* **9**(2), 149–192.

Walia, V., Byde, A. & Cliff, D. 2003. *Evolving Market Design in Zero-Intelligence Trader Markets*. Technical report, Hewlett-Packard Research Laboratories, Bristol, England.

Wilensky, U. 2002. Modeling nature's emergent patterns with multi-agent languages. In *Proceedings of EuroLogo-2002*, Futschek, G. (ed.). Austrian Computer Society.

Wilensky, U. & Resnick, M. 1999. Thinking in levels: a dynamic systems perspective to making sense of the world. *Journal of Science Education and Technology* **8**(1), 3–19.

Wilkinson, D. 2002. The intersection of learning architecture and instructional design in e-learning. In *e-Technologies in Engineering Education: Learning Outcomes Providing Future Possibilities*. Lohmann, J. R. & Corradini, M. L. (eds). `http://services.bepress.com/eci/etechnologies/33`

Wooldridge, M. 2002. *An Introduction to Multi-agent Systems*. John Wiley and Sons.

Wooldridge, M. & Jennings, N. R. 1995. Intelligent agents: theory and practice. *Knowledge Engineering Review* **10**(2), 115–152.