Exhibit at the International Conference on Intelligent Tutoring Systems, ITS'98, San Antonio, Texas, 1998.

The Disciple Learning Agent Shell and a Disciple Test Generation Agent

Gheorghe Tecuci, Harry Keeling, Tomasz Dybala, Kathryn Wright, David Webster

Department of Computer Science
MSN 4A5, George Mason University
4400 University Dr., Fairfax, VA 22030-4444
phone: (703) 993-1722, fax: (703) 993-1710, email: tecuci@gmu.edu

Abstract

The Disciple Learning Agent Shell is a tool for developing intelligent agents where an expert teaches the agent how to perform domain-specific tasks in a way that resembles the way the expert would teach an apprentice, by giving the agent examples and explanations as well as by supervising and correcting its behavior. Disciple can be used by educators to build educational agents that will assist them in various ways. Such an educational agent that was built with Disciple generates history tests for middle school students, assisting the teacher in the assessment of students' understanding and use of higher-order thinking skills. The agent can also provide intelligent feedback to the student in the form of hints, answer and explanations. The test generation agent and the process of building it are presented in (Tecuci and Keeling, 1998). We propose to demonstrate both the Disciple shell and the test generation agent.

1. Disciple Learning Agent Shell

Disciple is an apprenticeship, multistrategy learning approach for developing intelligent agents where an expert teaches the agent how to perform domain-specific tasks in a way that resembles the way the expert would teach an apprentice, by giving the agent examples and explanations as well as by supervising and correcting its behavior. The central idea of the Disciple approach is to facilitate the agent building process by the use of synergism at several levels. First, there is the synergism between different learning methods employed by the agent. By integrating complementary learning methods (such as inductive learning from examples, explanation-based learning, learning by analogy, learning by experimentation) in a dynamic, task-dependent way, the agent is able to learn from the human expert in situations in which no single strategy learning method would be sufficient. Second, there is the synergism between teaching (of the agent by the expert) and learning (from the expert by the agent). For instance, the expert may select representative examples to teach the agent, may provide explanations, and may answer agent's questions. The agent, on the other hand, will learn general rules that are difficult to be defined by the expert, and will consistently integrate them into its knowledge base. The Disciple approach is implemented in a tool, called Disciple Learning Agent Shell, that significantly reduces the involvement of the knowledge engineer in the process of building an intelligent agent.

The Disciple Learning Agent Shell is implemented in Common LISP, and runs on Macintosh. It consists of four main domain independent components shown in the light gray area of Figure 1. They are:

- a **knowledge acquisition and learning component** for developing and improving the knowledge base, with a domain-independent graphical user interface;
- a **basic problem solving component** which provides basic problem solving operations;
- a **knowledge base manager** which controls access and updates to the knowledge base;
- an empty **knowledge base** to be developed for the specific application domain.

The two components in the dark gray area are the domain dependent components that need to be developed and integrated with the Disciple shell to form a customized agent that performs specific tasks in an application domain. They are:

- a **specialized problem solver** which provides the specific functionality of the agent;
- a domain-specific **graphical user interface**.

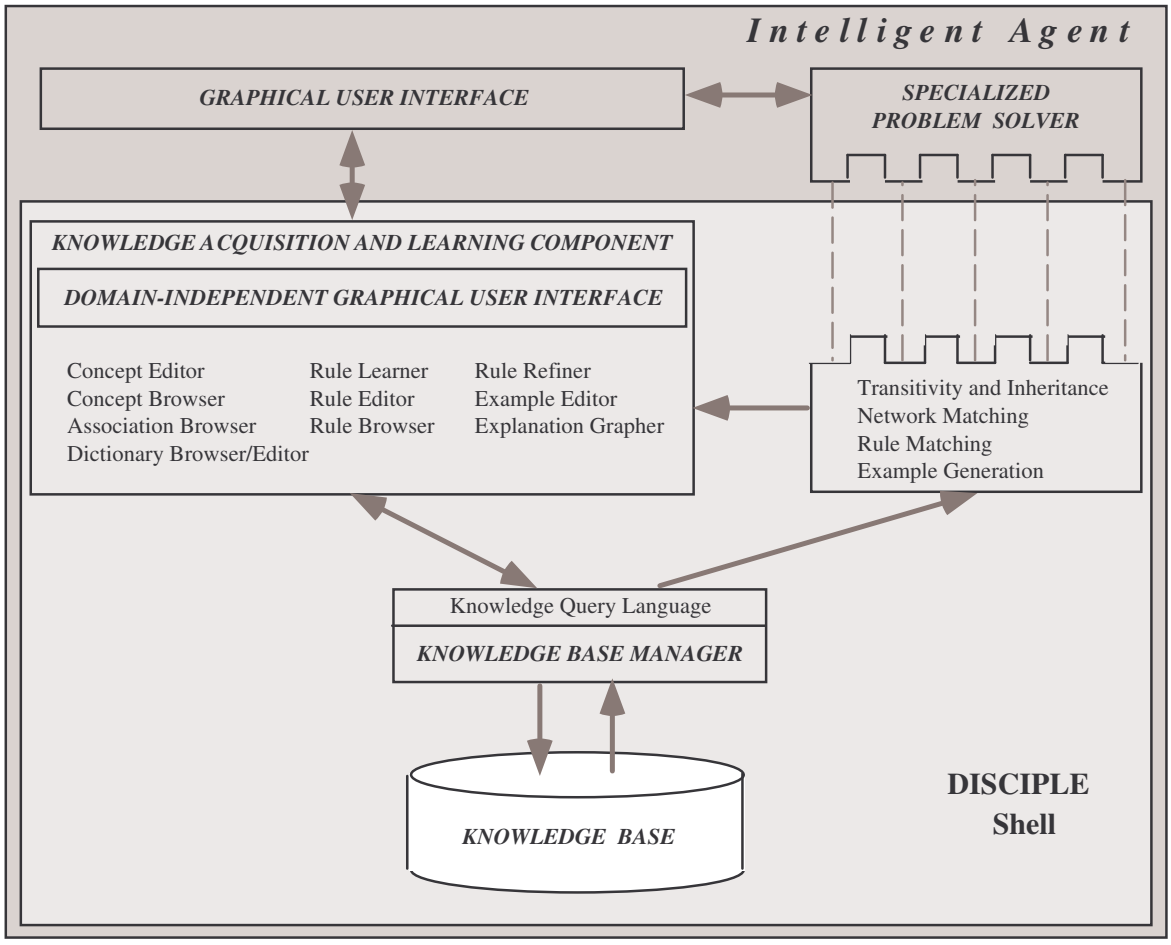

Figure 1: Architecture of the Disciple Shell

In the case of the test generation agent built with Disciple, the specialized problem solver is the test generator that also builds and maintains a student model. Two domain specific interfaces were built. One was built to facilitate the communication between the history expert/teacher and the agent. The other was built to facilitate communication between the agent and the students.

We will demonstrate the use of the Disciple Learning Agent Shell to build an educational agent that generates history tests for students. We will also demonstrate the use of the test generation agent.

2 Using the Disciple shell to build the test generation agent

First, the knowledge engineer customizes the general Disciple learning agent shell into a specific learning shell for test generation, by defining several interface modules. One is the Source Viewer that allows the agent to access and display historical sources from a multimedia database. Another is a Customized Example Editor that allows the expert to give examples of test questions using natural language templates. Yet another is a test interface that allows the agent to display test questions. The knowledge engineer also builds a test generation engine that relies on example generation, a basic problem solving operation supported by the Disciple shell (see Figure 1).

After the learning shell for test generation has been built, the history expert or teacher interacts with it to develop its initial knowledge base and to teach it higher-order thinking skills such as how to judge the relevance of a historical source to a given task. The expert starts by choosing a historical theme (such as Slavery in America) for which the agent will generate test questions. Then the expert identifies a set of historical concepts that are appropriate and necessary to be learned by the students. The expert also identifies a set of historical sources that illustrate these concepts and will be used in test questions. All these concepts and the historical sources are represented by the history expert/teacher in the knowledge base of the assessment agent, by using the various editors and browsers of the customized learning agent shell (see Figure 1). This initial knowledge base of the agent is assumed to be incomplete and even possibly partially incorrect and will be improved during the next stages of knowledge base development.

Once the initial knowledge base is built, the expert teaches the agent higher-order thinking skills such as judging the relevance of a source to a given task. During this process the agent learns reasoning rules and also extends and improves the semantic network of objects and concepts.

To teach the agent how to judge the relevance of a source to a given task, the history expert interacts with the Customized Example Editor to provide an example of a task and a source relevant to this task. The agent tries to understand why the source is relevant to the given task by using various heuristics to propose plausible explanations from which the expert has to choose the correct ones. The expert may also provide additional explanations. Based on the selected explanations and on the initial example the agent automatically generates an initial rule. Next, the agent generates examples analogous to the initial example. These will be similar tasks and sources that the agent assumes to be relevant. Each generated example is shown to the teacher who is asked to accept it as correct or to reject it, thus characterizing it as a positive or a negative example of the rule. These examples are used to refine the rule. Other rules for judging the relevance of historical sources to other types of tasks are learned in a similar fashion.

The test generation agent is implemented in Common LISP, except for the interface which is implemented in JAVA, to allow the agent to run on different platforms.

3. The test generation agent for higher order thinking skills in history

The built agent generates history tests to assist in the assessment of students' understanding and use of higher-order thinking skills, such as the evaluation of historical sources for relevance. To motivate the middle school students, for which this agent was developed, and to provide an element of game playing, the agent employs a journalist metaphor. It asks the students to assume the role of a novice journalist who has to complete assignments from the Editor. One assignment could be to write an article on the experience of African American women during the Civil War. Within this context, the students are given source material and asked various questions that would require them to exercise higher-order thinking skills in much the way journalists do when they complete their assignments and prepare stories for publication.

The agent dynamically generates a test question, based on a student model, together with the answer, hints and explanations. Figure 2 shows such a test question generated by the agent. The student is given a task, a historical source and three possible reasons why the source is relevant to the task. He/she has to investigate the source and decide which reason(s) account for the fact that the source is relevant to the task. The student is instructed to check the box next to the correct reason(s). No check means that the reason is not correct. In the presented example, the student has correctly indicated that only reason "C" explains why the source 'Christmas in Virginia' is relevant to the task. The agent has displayed feedback that evaluates the student's answer.

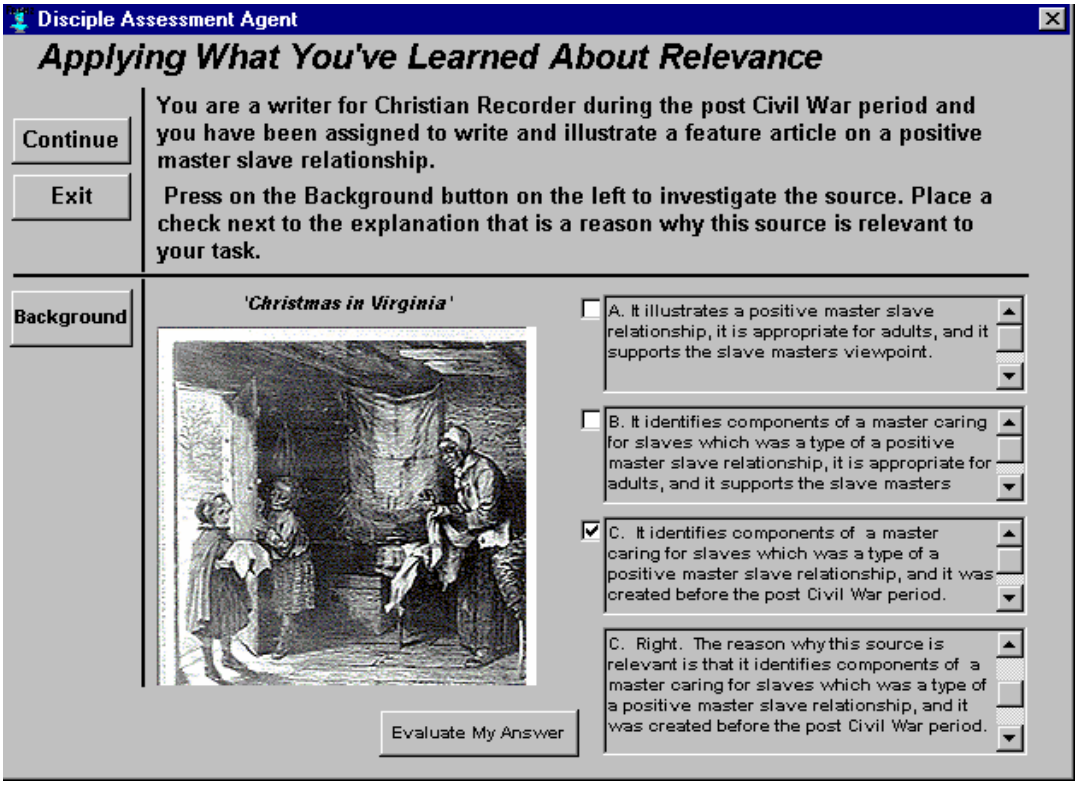

Figure 2: Why relevant test question with feedback for right answer*

* Picture reproduced from LC-USZ62-30813, Library of Congress, Prints & Photographs Division, Civil War Photographs

The agent has two modes of operation: final exam mode and self-assessment mode. In the final exam mode, the agent generates an exam consisting of a set of test questions of different levels of difficulty. The student has to answer one test question at a time and, after each question, he or she receives the correct answer and an explanation of the answer. In the self-assessment mode, the student chooses the type of test question to solve, and will receive, on request, feedback in the form of hints to answer the question, the correct answer, and some or all the explanations of the answer. That is, in the self-assessment mode, the agent also tutors the student. The test questions are generated such that all students interacting with the agent are likely to receive different tests even if they follow exactly the same interaction pattern. Moreover, the agent builds and maintains a student model and uses it in the process of test generation. For instance, to the extent possible, the agent tries to generate test questions that involve historical sources that have not been investigated by the student, or historical sources that were not used in previous tests for that student. The number of test questions that the agent can generate is of the order of 10^5.

Acknowledgments

This research was supported by the DARPA contract N66001-95-D-8653, as part of the Computer-Aided Education and Training Initiative, directed by Kirstie Bellman. Partial support was also provided by the NSF grant No. CDA-9616478, as part of the program Collaborative Research on Learning Technologies, directed by Caroline Wardle.

Reference

Tecuci G. and Keeling H. (1998), Developing Intelligent Educational Agents with the Disciple Learning Agent Shell, submitted to ITS'98.