RoboTA: An agent colony architecture for supporting education

Kenneth D. Forbus
Institute for the Learning Sciences
Northwestern University
1890 Maple Avenue
Evanston, IL 60201, USA
+1-847-491-7699
forbus@ils.nwu.edu

Sven E. Kuehne
Institute for the Learning Sciences
Northwestern University
1890 Maple Avenue
Evanston, IL 60201, USA
+1-847-467-1976
skuehne@ils.nwu.edu

1. ABSTRACT

Resources in education and training are always limited, and instructors almost never have enough time to spend on the problems of their students. We believe that using a distributed agent architecture to provide coaches that operate outside software installed on the student's machine can provide valuable advantages in education and training. This paper describes RoboTA, an architecture for a colony of distributed agents aimed at supporting instructional tasks.

1.1 Keywords

Agent architectures, software agents, education, distributed systems

2. INTRODUCTION

In education and training there are never enough instructors to go around. Resources are always limited, so using computers to complement human instructors has been a long-standing motivation for research on AI in education. Although software coaches typically are not as good as the best human instructors, the ability to provide feedback on the student's schedule rather than the instructor's can make them a useful component in an educational system. RoboTA is a colony architecture for software coaches in educational environments. A RoboTA agent colony has some specialized agents, plus agents written for specific courses. Students will interact with RoboTA via email. Instructors will interact with RoboTA via email and private web sites. We see the key advantages of the RoboTA architecture as: 1.) Exploiting communication media that students are already comfortable with, 2.) A simplified creation of new software coaches, 3.) A simplified evolution of coaching services, 4.) Data recording for coaching and improvement of service, and 5.) Providing automated grading services.

Development of the RoboTA colony architecture has been driven by need. One of our research projects involves creating *articulate virtual laboratories* [1] [2]. One of our prototypes, CyclePad, helps students learn engineering thermodynamics. The AVL approach is design-oriented, on the grounds that students find design tasks highly motivating and such tasks require them to learn the fundamentals more deeply. CyclePad is now used by a number of universities on an experimental basis, some of them with large populations of student users (e.g., around 60 students per year at the US Naval Academy) and some of them quite distant (e.g., University of Queensland in Brisbane, Australia).

Providing support and gathering data from a diverse set of remote sites has proven to be a challenging problem. Also, our desire to provide improved coaching must be balanced with our user community's need for stability and keeping runtime systems small enough to work well on the PC's our student populations have. The ability to evolve our coaching software while giving our users the stability they desire, along with data gathering, were the initial driving considerations for RoboTA. We soon realized that moving from a single-agent model to a colony model, where some specialized agents provided infrastructure services for the rest, should enable the system to support multiple projects with only a small amount of additional complexity.

3. THE ROBOTA ARCHITECTURE

If we were only supporting CyclePad, a single server-based architecture would be adequate. However, we are fielding

a second AVL, designed to help high-school students learn controls theory, and other educational software as well. There are also aspects of our own classes that could benefit from automated checking of student work and providing anytime advice. Consequently, we designed RoboTA as an *agent colony*, so that new agents providing new services could easily be added. A RoboTA colony has two kinds of agents: A central server process (the *PostOffice*) and course or application-specific agents (*TA agents*). A colony is built from a toolkit that provides the PostOffice code, with hooks for customization, and a TA agent shell which provides the communication infrastructure for courses or software to be supported. Figure 1 shows an overview of the architecture of RoboTA.

The PostOffice handles all communication with students, dispatching each message to the appropriate agents. A transaction begins with a student sending email to the PostOffice. A set of *delivery rules* determines which agent(s) should handle each message received[1]. If no agent was found the message is returned to the student noting this fact.

The PostOffice communicates with TA agents via KQML messages over TCP/IP socket connections. TA agents can thus be distributed over a network, with new servers added as needed. It is often advantageous to have the agent running on a machine under the control of the agent's author[2]. This makes modifying the agent code much easier, as it makes no demands on the PostOffice administrator. The TA agent responds to the student's request, often generating a reply. This reply is sent back to the PostOffice via a KQML message, and the PostOffice then generates email with this reply to respond to the student.

Adding a TA agent to a RoboTA colony requires two steps. First, the PostOffice administrator must add delivery rules for the TA. Central administration of the delivery rules is necessary to ensure the integrity of the system and its data. Adding a new TA agent is, however, an infrequent task.

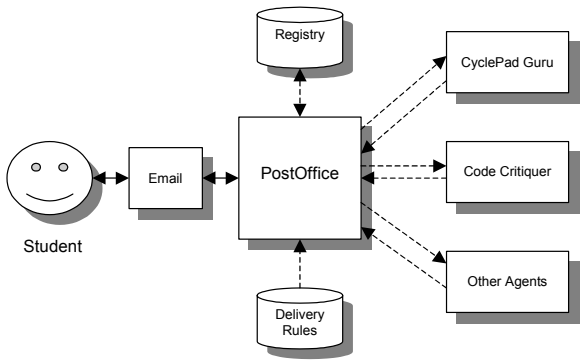

Figure 1: The RoboTA Architecture

Second, the TA agent must register its address with the PostOffice. This step relies on a password agreed upon during the first step, but otherwise is done without intervention. This facilitates moving an agent to another machine if its host fails. The establishment of these links, plus caching them in the PostOffice registry, is handled by the communication infrastructure in the RoboTA agent toolkit.

4. CURRENT STATUS

RoboTA is currently in active use for providing assistance and feedback to the users of CyclePad. The built-in email facility allows users of CyclePad to send in their design and get feedback from a design coach (the CyclePad Guru) that runs as a RoboTA agent.

As another member of the agent colony, we will also integrate a code critiquer for Scheme into RoboTA to assist undergraduate programming classes. This critiquer will provide feedback on good programming style. This will allow us to offer a convenient automated service to students, instead of manually feeding submitted programs into the critiquer and then mailing back the results.

5. ACKNOWLEDGEMENTS

We thank John Everett for feedback. This research was supported by the National Science Foundation and the Office of Naval Research.

6. REFERENCES

[1] K. Forbus and P. Whalley, *Using Qualitative Physics to Build Articulate Software for Thermodynamics Education*, Proc. 12th National Conference on Artificial Intelligence, Vol. 2, AAAI Press/MIT Press, Cambridge, Mass., 1994, pp. 1175-1182.

[2] K. Forbus, *Qualitative Physics to Create Articulate Educational Software*, IEEE Expert Intelligent Systems & Their Applications, Vol. 12, No. 3: May/June 1997, pp. 32-41.

[1] Allowing multiple agents to handle a message supports handling complex chores by multiple agents. A student request to sign up for a course might be processed by separate agents who maintain class lists, registration information and send out the reading material or software for the course. This capability is not yet used.

[2] For example, we have had requests from other instructors at Northwestern who want to add TA agents for their courses. In such cases we provide them the RoboTA toolkit and they provide and maintain the necessary hardware upon which their TA agent is run.