Designing Human Benchmark Experiments for Testing Software Agents

Robert D. Grant, David DeAngelis, Dan Luu, Dewayne E. Perry
Empirical Software Engineering Lab
The University of Texas at Austin, Austin, TX 78712
Email: {bgrant,deangelis,luu,perry}@mail.utexas.edu

Kathy Ryall
Advanced Information Technologies
BAE Systems, Burlington, MA 01803
Email: kathy.ryall@baesystems.com

***Abstract*—**

Background: **Software agents are becoming increasingly common in the engineering of software systems. We explore the use of humans in creating benchmarks for the evaluation of software agents. In our case studies, we address the domain of instructable software agents (e-students) as proposed by the Bootstrapped Learning project [Oblinger, 2006].**

Aim: **Our aim is to define and refine requirements, problem solving strategies, and evaluation methodologies for e-students, paving the way for rigorous experiments comparing e-student performance with human benchmarks.**

Method: **Little was known about what factors would be critical, so our empirical approach is exploratory case studies. In two studies covering three distinct groups, we use human subjects to develop an evaluation curriculum for e-students, collecting quantitative data through online quizzes and tests and qualitative data through observation.**

Results: **Though we collect quantitative data, our most important results are qualitative. We uncover and address several intrinsic challenges in comparing software agents with humans, including the greater semantic understanding of humans, the eidetic memory of e-students, and the importance of various study parameters (including timing issues and lesson complexity) to human performance.**

Conclusions: **Important future work will be controlled experiments based on the experience of these case studies. These will provide benchmark human performance results for specific problem domains for comparison to e-student results.**

I. Introduction

Software agents are becoming increasingly common in the engineering of software systems. These agents are generally intended to be autonomous and independent, and may be specialized to a particular domain or required to function in unforeseen domains. There are a variety of techniques for creating that autonomy; in this paper we target domain-independent human-instructable agents. Whatever the approach used, we should rigorously evaluate agent performance through empirical studies. In this research we explore the use of humans in creating benchmarks for the evaluation of software agents.

II. Study Aims

In this paper we present two exploratory case studies with human subjects wherein we define and refine requirements, problem solving strategies, and evaluation methodologies for instructable software agents (*e-students*) as proposed by the Bootstrapped Learning project [Oblinger, 2006]. The eventual goal is to directly compare e-student learning with human-student learning on "identical" curricula in controlled experiments.

We aimed to produce lessons and tests on which human students who scored less than 20% in pre-test could score at least 75-80% in post-test, indicating that learning occurred.

III. Related Work

A. The DARPA Bootstrapped Learning Project

Bootstrapped Learning is a novel approach to machine learning whose goal is to produce software agents that can be taught by human instructors in the same ways that humans instruct one another. These e-students could be trained by domain experts who are not necessarily skilled programmers; this is especially valuable for systems that benefit from being *field-trainable*, or specializable to a particular need by end users at a faster rate than is usually supported by a traditional software development lifecycle.

Two teams have been working in parallel as part of the DARPA BL project. Our group is part of the Evaluation Team, which also includes several other groups. The Evaluation Team is developing BLADE (Bootstrapped Learning Analysis and Curriculum Development Environment). This includes developing a framework to support BL research, sets of curricula across a variety of domains as testing vehicles for the e-student, and an evaluation of the e-student on both hidden and known domains. A separate Learning Team is developing the actual e-student, incorporating several learning strategies.

BLADE includes three agents, whose interactions and relationships are shown in Figure 1. A teacher agent serves as a proxy for an eventual human teacher, instructing and testing the e-student. The student agent is the embodiment of the e-student. The world agent serves as a proxy for a domain simulator. Over the first three phases of the BL program, the Evaluation Team has developed curricula in a variety of complex domains including Blocksworld [Berland and Perry, 2009], unmanned aerial vehicles (UAV), diagnosis tasks for an international space station (ISS), armored task force maneuvers (ATF), planning robotic arm movements, and a hidden domain.

BLADE uses IL (InterLingua) and ITL (InteracTion Language), developed specifically for the BL project [Oblinger, 2006], to pass messages between agents in the BLADE

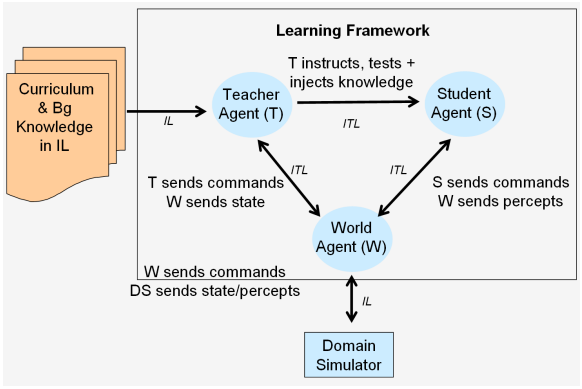

Fig. 1. The BLADE Framework

framework. It uses an automated teacher in place of a human teacher to improve testing scalability and reproducibility. Part of the Evaluation Team's research is to explore how best to incorporate a human teacher. In a parallel effort, the Evaluation Team is developing a tool to support human-/e-student instructional interactions, in part informed by the evaluations described here.

B. Human Learning and Teaching

Like its human counterpart, an e-student assumes its instructor possesses all relevant capabilities, and its goal is to learn using the same instruction methods used between humans. As part of the Evaluation Team, we are not allowed access to e-student implementation details to ensure our benchmarks are unbiased. We know that the learners are designed to be domain-independent, and that they are specialized to particular Natural Instruction Methods (NIMs) rather than particular problem domains.

The area of *computer tutoring* can be seen as an inverse problem to what we are investigating. In particular, the area of *teachable agents* bears some surface similarity to BL. In this field, human students teach learning agents in order to improve their own understanding of concepts ("Learning by Teaching"). One example is the "Betty's Brain" system [Davis et al., 2003]. However, in these systems the importance is placed on how well the human instructor learns, not on the capabilities of the learning agent.

C. Human Case Studies

Our human studies use well known techniques from behavioral research, as covered in standard texts such as Rosenthal and Rosnow [1991] and Yin [2008]. In particular, since little was known about what factors would be critical, our empirical approach is that of exploratory case studies. An exploratory case study is the best approach when little is understood about the subject under study [Yin, 2008]. The intent is to build a deeper understanding of the phenomena in question and to formulate the beginnings of a corresponding theory that can be tested, revised, and expanded with further empirical studies.

Our group conducted a "Phase I" study prior to those described in this paper [Berland and Perry, 2009]. In this case study, each of five human teachers (HTs) attempted to teach an e-student to construct a simple doorway out of blocks. This doorway was a simple structure of two stacks of square blocks topped with a lintel (a long block). Perry et al. used a Wizard of Oz (WOz) [Dahlbäck et al., 1993] style methodology, where the human teachers' natural-language instructions were translated into precise terms (IL) for the e-student by a human interpreter. The HTs were asked to consider teaching to the level of a bright two-year-old. This study informed later studies in the ways human teachers would try to teach an e-student and in the limitations of current e-student understanding.

IV. The Studies

We performed two studies, one in Summer 2009 and one in Summer 2010, which we respectively refer to as "Phase II" and "Phase III". Our overarching goal is to mimic the e-student context as closely as possible in the human studies to ensure the comparison is valid. In the Phase II study, the focus is on creating the setup and protocols; in Phase III, we refine and extend both the lab setup and the experimental protocols.

A. Natural Instruction Methods

We present the curriculum to students via three different *Natural Instruction Methods (NIMs)*: teaching by telling, teaching by example, and teaching by feedback. Respectively, these consist of utterances emitted by a teacher, examples performed by a teacher in a simulator, and instructions for a student to apply techniques in a simulator with teacher feedback. We abbreviate these lesson types as T, E, and F. In Phase II, we gave each human student all three lesson types, with the option to skip. In Phase III, we tested some students with all three NIMs, and other groups with only one or two NIMs. We also allowed a small group of students to ask questions about the curriculum. We refer to the students who received all three instruction types as the *baseline*.

B. The Hidden Domain

We cannot publish the subject matter of the hidden domain until e-student testing has taken place, but we can discuss its general nature and the mechanisms human students use to interact with it. Human students learn about and perform tasks in the domain through the use of a Java simulator with a graphical user interface (GUI) in which the user is informed of the state of the simulation and may take various actions to manipulate that state.

C. Phase II: Initial Study Design

Our first design was a direct analog of the relationship between an automated teacher and an e-student. Since our objective was to evaluate the curriculum and not the subjects, a major concern was preventing the human teacher from unconsciously providing extra-curricular information to a student through facial expressions, gestures, tone, etc. In this design there was one teacher, one student, and at least one observer per session. The teacher and student were each provided two laptops and one external monitor and positioned on desks facing one another, separated by a screen (Figure 2). One laptop on each side was used for instant messaging between

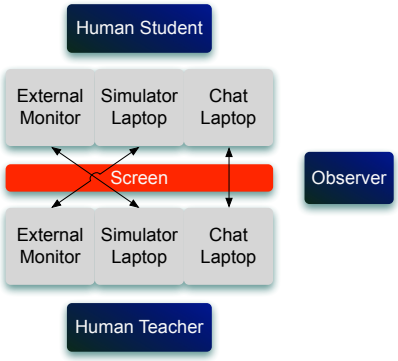

Fig. 2. Phase II Initial Setup: Instructor-led Curriculum

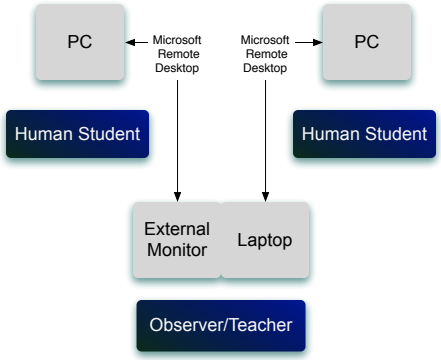

Fig. 3. Phase II Final Setup: Self-paced Curriculum

teacher and student, and the other was used for manipulation of the hidden domain simulator. Each side's external monitor was connected to the hidden domain simulator laptop on the other side of the separating screen. This allowed a student to see the teacher's example usage of the simulator and allowed the teacher to observe a student's simulator practice and tests.

The student and teacher were only allowed to communicate through the electronic means provided. The student was allowed to talk to the observer about practical issues like need for breaks, failures of equipment or software, or desire to withdraw from the study. For teaching, the human teacher simply typed transliterations of the electronic curriculum into the chat window and performed curriculum examples in the simulator on the corresponding laptop. We used a screen-drawing tool to allow the teacher to emphasize certain areas of the screen by circling and underlining. The student was only allowed to ask the teacher to skip or repeat a lesson; no other communication was allowed, as there is none between the e-student and automated teacher.

D. Phase II: Problems with Initial Design

There were several practical problems with this initial setup and curriculum. First, because of the restricted communication method, there was no way for a teacher to tell if a student had seen and understood an utterance or demonstration, so the teacher could not pace the teaching correctly. Also, the teacher's job of typing utterances and giving demonstrations was time-consuming, error-prone, and there was no protocol allowing the teacher to report errors or redo instructions.

These minor issues were easy to fix, but there were deeper problems with teaching human students using a direct translation of the e-curriculum. Mainly, this method was very slow, and we were overrunning our allotted study time of four-hours-per-person by a large margin. Also, both the method of communication and the phrasing of instructions were sometimes awkward, frustrating, and misleading for human students.

To solve these problems, we thought carefully about human- and e-student differences and considered how we could improve the human testing process while maintaining an acceptable equivalence to the e-student curriculum. One major issue is that e-students have the advantage of eidetic memory. This means that an e-student is able to recall any given lesson exactly, unlike most human students. On the other hand, human students have a much deeper understanding of language. This can sometimes lead a human student to glean more useful information from a given instruction than an e-student could, but it also sometimes confuses a human student when an instruction has multiple possible meanings, or when an instruction is unnaturally phrased (though strictly correct).

E. Phase II: Final Study Design

Our main modification that made human testing feasible was making the curriculum self-paced. Instead of a human teacher feeding every line of curriculum to a student and demonstrating simulator usage manually, we formatted the curriculum as PowerPoint slides with instructions and accompanying figures. For the feedback lessons, we provided the student with instructions on procedures to try in the simulator and what the outcome should look like if the procedure was performed correctly; we called this the "choose-your-own-adventure" style.

This eliminated several complexities in our initial laboratory design. Instead of two laptops and an external monitor for both student and teacher, a student could now go through the curriculum on a single laptop, and a teacher/observer could watch over a subject's shoulder or through remote desktop software (see Figure 3). We no longer needed the awkward restricted communications channel, and we no longer needed to worry about issues of pacing or instructor error. This format had the additional advantage of allowing us to run multiple students in parallel with only one teacher/observer.

We addressed the issue of the e-student's eidetic memory by allowing human students to take notes and review previous lessons at any time.

The issue of awkward and ambiguous language was harder to solve; we overcame it by beta testing our curriculum with several students and changing the offending parts based on feedback. We ended up with a more "natural" expression of the instructions for human testing despite the small risk of deviating from the electronic curriculum. The direct translation from the formal language of the e-curriculum was simply too confusing for human students. Moreover, when we encountered semantically laden terms (such as in the names of lessons), we substituted neutral language–for example "Lesson Blue".

While self-pacing considerably decreased the study duration, the curriculum was still too long for a human student to

complete in four hours. In our final version, several curriculum sections were merged for readability, and each subject was tested on a subset of the final questions given to the e-student. The mean post test score for Phase II was 91%.

F. Phase III: Initial Study Design

For Phase III, we increased the difficulty and complexity of tasks given to learners. For example, in Phase II precise terms were used in the vocabulary of the lessons, and students were given certain pieces of basic information about each lesson, including which NIMs were being used and the names of the procedures being taught. This initial approach was a result of some of the lessons learned in Phase I [Berland and Perry, 2009], where the human teachers had to be very precise in their bottom-up instructions in order for the e-student to understand the lessons being taught.

To add realism to the test scenarios, we also configured the simulator to advance through states in real-time rather than allowing the subject to manually advance the state.

G. Phase III: Problems With Initial Design

Subjects' scores were unexpectedly low with the real-time test clock. We hypothesize that this was for two reasons:

a) Training vs. Education: Though the overall test-scenario time was greater than in Phase II (8 minutes rather than 5 minutes), the critical time-window in which subjects were required to perform each task was shorter (1 minute rather the entire 5 minutes of Phase II). Students' scores improved when the real-time constraint was removed, so we think this indicates that success in these real-time scenarios is more a matter of training (skill gained through repetitive practice) than education (knowledge gained through learning). We do not know if this will be an issue for e-students.

b) Boredom: Since subjects were not allowed to manually advance the simulator past states in which nothing occurs, we observed them losing focus while waiting through these states. Subjects were clearly less attentive and energetic when critical simulation events came to pass.

Also, one scenario seemed to be too difficult; it was a scenario where multiple actions were required instead of just one, and many students missed one of the actions.

H. Phase III: Final Study Design

In our final version of the study, we returned to allowing subjects to advance the simulation by hand. Average final test scores for the baseline jumped from 57% to 81%, validating our hypothesis that a large part of the score reduction was due to the real-time clock.

Additionally, we greatly streamlined and automated our testing setup. Whereas in Phase II we could test at most two subjects at a time, in Phase III, through automation, we increased our capacity to six at once. We provided each student with their own workstation in our testing lab, as shown in Figure 4.

Each workstation mounts our Network Attached Storage (NAS) device containing a specially generated curriculum

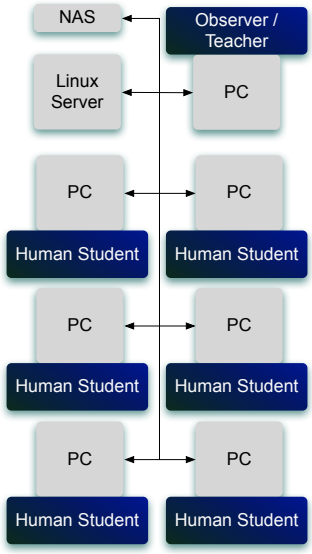

Fig. 4. Phase III Setup

folder for each station. The corresponding curriculum folder is linked on the desktop of each machine, making all study materials easily accessible to each subject.

We also mount the NAS on our Linux server. This allows us to run curriculum-generating scripts on the server which automatically create and distribute new materials to each workstation in preparation for each study group. These scripts handle tasks such as randomizing pre- and post-test selection and ordering and allow us to generate curricula for special treatment groups such as restricted-NIM treatments.

All study materials are kept in a Subversion repository on our Linux server. When study materials are distributed to subject workstations, our scripts also dump metadata (such as Revision Number) into the distribution. Thus, when we back up a subject's study materials after a session, the backup includes information on the exact version of the study a subject received.

To ensure subjects follow proper protocol, study supervisors monitor subject progress by direct observation and through monitoring scripts set up on the server. In Phase III we have also implemented automatic grading of subject tests, and we have the ability to playback tests if there is any question as to what actions were taken.

I. Participant Selection, Scheduling, and Procedure

Study participants were volunteer respondents to mass emails sent to graduate-student mailing lists at The University of Texas at Austin, including Engineering, History, and Public Policy mailing lists. Participants were selected on a first-come first-served basis, and those who qualified were offered $75 if they completed the study or $7.50 per hour if they withdrew or were unable to finish. Participants were told the study would take around three hours, and that they would have a maximum of four hours to complete the study. Participants were assigned testing times based on their preference and availability. Before students came to the lab they were provided with a broad overview of how the study would proceed and basic instruc-

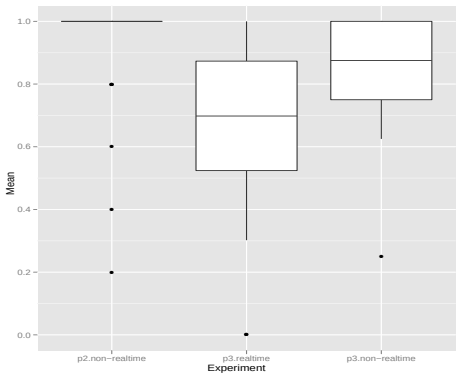

Fig. 5. Baseline Post-Test Means

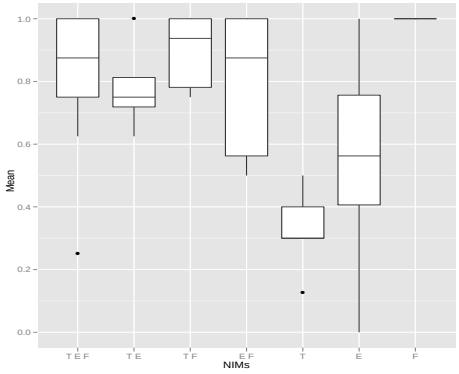

Fig. 6. Post-Test Means, By NIM

tions. The approximate schedule for a participant follows:

- (15 mins.) Introduction, background material presentation
- (15 mins.) Pre-test
- (3 hrs.) Self-paced lessons with web-based quizzes
- (30 mins.) Post-test

J. Quantitative Results

In all results we exclude students that scored above 20% on the pre-test. We do not separate subjects by major, as those in non-technical majors did as well as those in technical majors. The small group allowed to ask curriculum questions of the study supervisors asked no questions of note, so we do not separate them in these results.

In Figure 5, we compare three groups of baseline subjects: those in Phase II (p2.non-realtime), those in Phase III with a realtime test clock (p3.realtime), and those in Phase III without a realtime test-clock (p3.non-realtime). These groups contained 28, 12, and 19 subjects respectively. The mean post-test score for p3.realtime students was 57%; for the p3.non-realtime students, the mean was 81%, surpassing our goal of 80%. It is clear from the figure that our curriculum changes between the two Phase III groups improved scores across the board, not just the mean. The Phase III curriculum overall is more difficult than the Phase II curriculum, and our p3.non-realtime curriculum achieved our goal of 80% while being more discriminative than the Phase II curriculum.

In Figure 6, we show mean post-test score by NIM-set. The two-NIM subjects seemed to do almost as well as those given the all three. The subjects given the F NIM seemed to do the best of the restricted sets; there are several possible reasons why. This NIM is somewhat a combination of F and T lessons; in the self-paced F lessons, if a subject doesn't follow the correct procedure, the subject is given a "by telling" description of what should have been done. This is also the only NIM where subjects get any practice with the simulator before the post-test.

V. Lessons Learned

- Human- and e-students differ in fundamental ways that make it difficult to create analogous contexts without providing one side with undue advantages.
- Seemingly insignificant semantic details are critically important in attempting to provide analogous contexts.
- The self-pacing mechanism for human teaching and learning has proved essential in establishing the e-student benchmarks.
- Increased automation of lesson structures was essential for the expansions of Phase III.
- The issue of training versus education raised in Phase III is a fundamental one, and it needs to be explored in depth. Results from Human-Computer Interaction (HCI) may be relevant to this question; specifically, work on models of human task-performance such as the Human Model Processor [Card et al., 1986] and GOMS [John and Kieras, 1996].

Acknowledgements

This research was sponsored by AFRL under contract FA8650-07-C-7722. Special thanks to our colleagues at BAE Systems, Cycorp, Inc., Sarnoff Corporation, Stottler Henke Associates, Inc., and Teknowledge Corporation for their collaboration in developing the various BL curricula and testing materials, to Matthew Berland for early work on this project, and to study participants at The University of Texas at Austin.

References

D. Oblinger, "Bootstrapped learning: Creating the electronic student that learns from natural instruction," AAAI Briefing, 2006. [Online]. Available: http://www.darpa.mil/ipto/programs/bl/docs/AAAI_Briefing.pdf

M. Berland and D. E. Perry, "Novice human teachers of a virtual toddler: A case study," The University of Texas at Austin, Tech. Rep., 2009. [Online]. Available: http://users.ece.utexas.edu/~perry/work/papers/090123-MB-blexp1.pdf

J. Davis, K. Leelawong, K. Belynne, B. Bodenheimer, G. Biswas, N. Vye, and J. Bransford, "Intelligent user interface design for teachable agent systems," in *Proceedings of the 8th international conference on Intelligent user interfaces*. ACM, 2003, pp. 26–33.

R. Rosenthal and R. Rosnow, *Essentials of behavioral research: Methods and Data Analysis*, 2nd ed., ser. McGraw-Hill Series in Psychology. McGrawHill, New York, 1991.

R. Yin, *Case study research: Design and methods*. Sage Publications, Inc, 2008.

N. Dahlbäck, A. Jönsson, and L. Ahrenberg, "Wizard of oz studies – why and how," *Knowledge-Based Systems*, vol. 6, no. 4, pp. 258 – 266, 1993, special Issue: Intelligent User Interfaces.

S. Card, T. Moran, and A. Newell, "The model human processor-An engineering model of human performance," *Handbook of perception and human performance.*, vol. 2, pp. 45–1, 1986.

B. E. John and D. E. Kieras, "The goms family of user interface analysis techniques: Comparison and contrast," *ACM Trans. Comput.-Hum. Interact.*, vol. 3, no. 4, pp. 320–351, 1996.