Teaching Abstract Data Type Semantics with Multimedia*

Glenn D. Blank, Edwin J. Kay, William M. Pottenger
Jeffrey J. Heigl, Soma Roy and Shreeram A. Sahasrabudhe
glenn.blank@lehigh.edu, pottenger@cse.lehigh.edu, **ejk0@lehigh.edu**
Department of Computer Science and Engineering
Lehigh University, Bethlehem, PA 18015

Abstract

CIMEL is a multimedia framework for Constructive, Inquiry-based E-Learning supplementing computer science courses. Within the CIMEL multimedia framework, we have developed new materials at different levels in computer science curricula, including a first semester course in computer science (CS0), a graduate level course on Object-Oriented Software Engineering, and an upper level undergraduate course in software engineering (SE). For the SE course, we developed a multimedia unit on Inheritance and dynamic binding and another on Abstract Data Types (ADTs). The latter formalizes the meaning of classes in connection with object-oriented design. In the past, upper level students have found it difficult to master this material from lecture and textbooks alone. Our premise is that multimedia will help students understand the material better, objectively, and also improve their design of actual ADTs to solve a problem. In this paper, we report on our most recent experiment, currently under way, evaluating multimedia to help teach Abstract Data Types in an SE course. The results are very promising.

Introduction

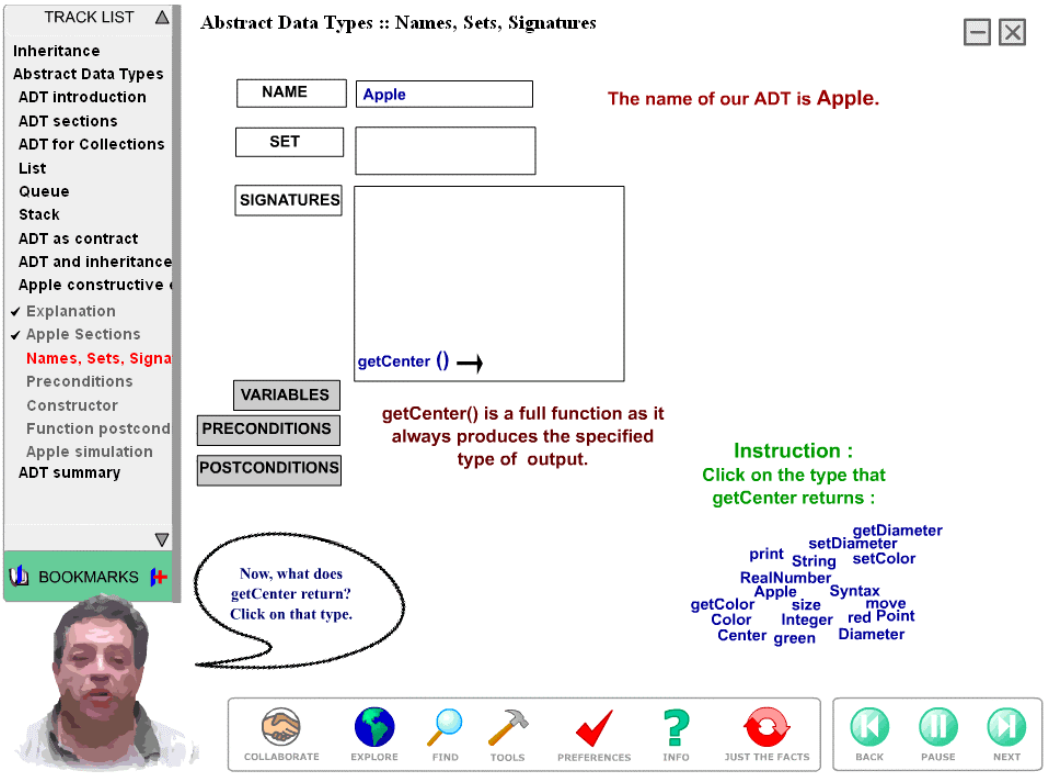

Figure 1: Screen Capture from *The Universal Computer* (as of March 2002)

*This project is funded by National Science Foundation (Grant Number EIA-0087977)

Figure 1 illustrates several features of the CIMEL multimedia framework:

- Multimedia personae model a community of learners and instructors. The personae include a professor (shown here), teaching assistant, a reference librarian, and two students. In addition to graphical images, they speak in audio and/or text boxes. These personae model students and teachers studying material together, working through interactive and constructive exercises, and suggest exploratory research on relevant topics using online information.
- A TRACKS LIST at left displays the content of a lesson as a sequence of screens. The menu uses check marks to show progress and highlights the current screen in red. An external XML file maintains a menu of screen titles, facilitating maintenance of course content.
- The icons at the bottom give learners access to various tools, including the COLLABORATE and EXPLORE (emerging trends) tools under development for this project, as well as the BlueJ and JavaEdit programming environments developed elsewhere.
- The PREFERENCES icon presents a panel of options letting the user adapt the environment according to his or her personal learning style, including turning text boxes or audio on/off, toggling auto-advance or wait for next page, setting the timing rate where there is no audio narration, etc. A user may change these settings at any time during these sessions and they will be recorded locally and on a network drive for the next session.
- A JUST THE FACTS mode lets users switch to viewing non-interactive content (text and graphics) presented in HTML pages. From there, one can switch back to rich media mode via hyperlinks anchored to the corresponding Flash page. There are also links to interactive screens, which remain in Flash.

Interactivity and constructive exercises

Interactivity is a key aspect of CIMEL content. Interactive quizzes and constructive exercises help students learn by doing. Personae provide feedback guiding a student through each exercise. In Figure 2, the TA persona changes expression and provides feedback hinting at what is wrong with a user choice in a multiple choice question.

Constructive exercises are much more complex, challenging a learner to build solutions to problems by dragging and dropping pieces of structures into place, incrementally. Figure 1 is a snapshot from a sequence of screens, in which a user gradually builds an abstract data type for Apple. In this exercise, the learner goes through a series of screens, incrementally building up an ADT specification for class Apple. The learner has already constructed the signatures section of Apple ADT by first choosing a member function, then in figure 1 selecting that function's arguments. Later on in the exercise, the learner builds the preconditions and postconditions for Apple ADT. Finally, the learner runs simulations running the functions, preconditions and postconditions, testing them for completeness. At each step, feedback helps the learner learn from mistakes as well as correct actions.

Inquiry-based exercises facilitate learning by doing research. For example, after studying ADT for collections, the screen shown in Figure 3 asks to the student to investigate how the design of similar ADTs in the most recent JDK. The following screen then asks a followup question designed to find out what the student has learned from the inquiry-based exercise.

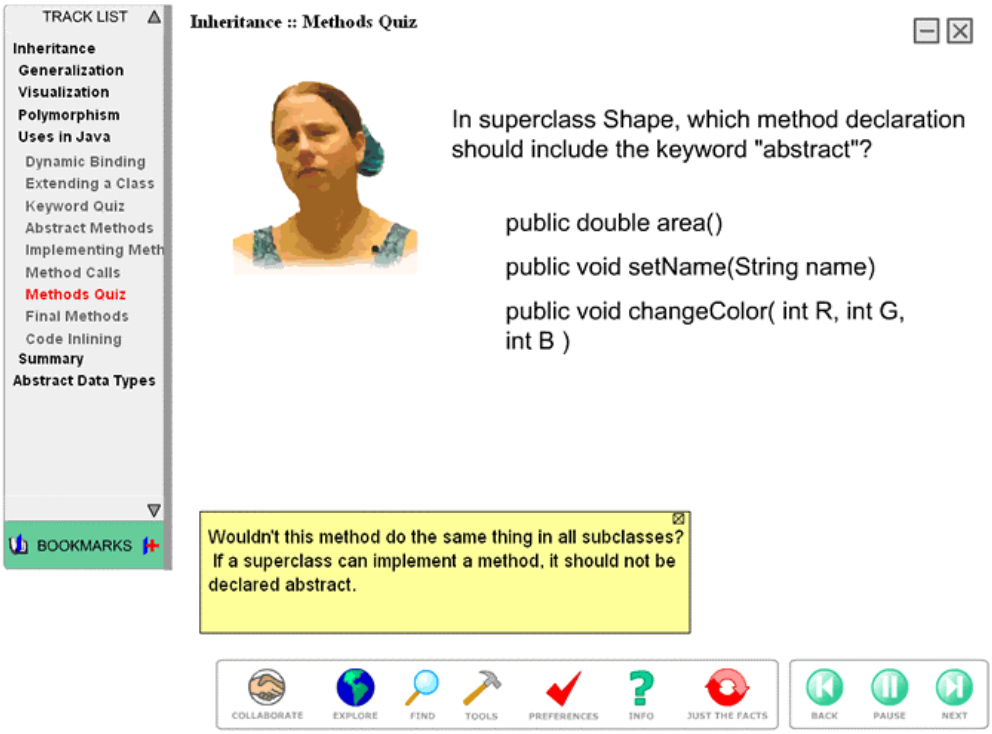

Figure 2: TA persona responds to a wrong choice in a multiple-choice question

Figure 3: The student is given resource links help with a solution to the question.

A text mining and visualization framework for the detection of incipient emerging trends in order will facilitate more elaborate inquiry-based learning exercises [1]. An emerging trend is a topic area for which one can trace the growth of interest and utility over time. The detection of

emerging trends in the course domain stimulates inquiry-based learning by providing an avenue of research into key developments in these related fields.

Evaluation Methodology and Results

In the fall of 2001, we conducted a study on abstract data types (ADTs) in a graduate level course on Object-Oriented Software Engineering (OOSE). We designed an experiment to determine whether multimedia actually improves learning, both in terms of objective knowledge and in terms of students' ability to perform a task designing ADTs for a sample problem involving several classes and inheritance. Mean scores on objective tests improved significantly, suggesting that the multimedia does indeed contribute to objective learning of this content [1]. However, the results for the results for task learning were less clear than the results for objective knowledge. These mixed results may have been due in part to the design of the experiment and in part to the design of the multimedia. Learning from our experience, we have improved both.

With respect to multimedia design, focus groups and usability surveys led us to improve both the user interface and content in many ways. For example, the just the facts mode of presentation of the content now provides an alternative to the Flash media-rich version of the content. Students in both first year and graduate level focus groups were roughly split about whether they liked media-rich techniques such as animated, graphical personae and audio narration in the alpha version. Those who liked the rich media included most of the women and non-majors, while those who found these media a bit annoying and wanted to get right to the nuts and bolts tended to be male and more experienced in computer science. Even students who prefer the media-rich presentation would like a way to review the material in a less media-rich form. This result suggested that it would be useful to provide HTML pages with the text and key graphics from Flash content pages, dropping the use of personae and audio. Interactive exercises remain as is. Finally, a summary with a checklist now appears at the end of the module, summarizing what one should consider in designing ADTs. Each item in the checklist includes a link back to the relevant screen, for review. Students in the OOSE focus group strongly agreed that a checklist would give them a better idea how ADTs were designed as well as a good review.

With respect to the experimental design, we switched to a 2x2 design, dividing a class of about 72 students in a upper level SE course into four groups: One fourth gets neither the multimedia nor lecture, a second gets just the multimedia, a third gets just a lecture, and the fourth gets both the multimedia and lecture. Before seeing anything, all four groups fill out an online pretest of 20 randomly ordered multiple-choice questions. All four groups get the same homework problem description (see http://www.cse.lehigh.edu/~cimel/eval/beta/fruitADTAssignment.htm). After completing the assignment, students fill out an online post-test with questions very similar but not identical to the pretest.

Our hypothesizes are 1) that there will the multimedia group will perform better than those getting nothing and 2) that the multimedia plus lecture will perform than the lecture only group, on both pretest to posttest improvement and task grades.

As of now we have completed the first half of our study, and so can report on the performance of students getting the multimedia and the assignment and those getting just the assignment. Students getting the multimedia did significantly better on the objective post-test of 20 multiple-choice questions (for which "I don't know" was one of four options), with a mean of 14.7 vs. 12.3, $F(1,26)=8.86$, $p<.01$. This result suggests that the multimedia does indeed contribute to

objective learning of this content. Students with the multimedia also did better on the ADT assignment, with a mean of 76.31 vs. 68.77, [F score coming soon!]. This result suggests that the multimedia also helps students design ADTs—though we might like to see even better performance.

In the second half our study we will study how well students do with a combination of multimedia and lecture vs. just lecture. We will also analyze the tracking data to see how students actually use the multimedia and whether that correlates with performance, learning styles, and/or demographics. The lead author plans to present complete results of these studies at the ECOOP workshop.

Conclusions

Within the CIMEL multimedia framework, we have developed and evaluated prototypes of materials for upper level and introductory courses in computer science. The preliminary results of our studies are promising.

After analyzing the results of the current study and usability surveys, we plan to make further refinements to the user interface, content and development environment for CIMEL content. This summer, we plan to investigate whether CIMEL multimedia can be adapted to target minority students as well as high school students, particularly from minority groups. This summer, a student from a historically minority institution, a secondary teacher from a local school district, and several high school students from the Students That Are Ready (S.T.A.R.) Academies will join the CIMEL development team. They will:

- Conduct focus group and/or web-based surveys of students and teacher(s) at historically minority college and at a local high school, showing them the current CIMEL user interface and sample content, as well as proposed changes, to determine their effectiveness for these audiences.
- Propose and design possible changes to the user interface, that might make it more suitable for black or Hispanic undergraduate students, in a graphical prototype (describing modified behaviors with notes). For example, this may involve the creation of a personae that may be more engaging to this audience. Propose changes to the content as annotations to current scripts.
- Implement proposed changes to user interface and content.

Once the user interface stabilizes, our efforts will shift to developing material for a CS0 course, which can be used either with a new textbook, *The Universal Computer: Introducing Computer Science with Multimedia* (Blank, Barnes and Kay, 2002) or as a stand-alone product. Since we are designing material for a wide range of students, we have designed a dynamic tracks interface that will let instructors and students manipulate what gets presented. For example, much of the ADT material may be too advanced for first year students; an introductory track on software engineering will include an excerpt from this material; the tracks menu will let instructors and students add (or subtract) material from the default CS0 track supplied by the developers.

More documentation about the CIMEL project is available at http://www.cse.lehigh.edu/~cimel. The beta version of the Inheritance and ADT units are available at http://cimel.cse.lehigh.edu (login as “guest” with password “guest”). We welcome beta testers at other institutions!

Related Work

There is a rich and growing literature of multimedia-based educational material. We compare CIMEL to a brief selection of comparable research efforts and systems.

ProgramLive [2] is a rich multimedia tutorial of the Java programming language. *ProgramLive*'s interface represents a notebook, within a browser. There are tabs to the side of the notebook display that can be used for the navigation of the material, as well as pop-up explanation of key terms. The CIMEL user interface also plays through a browser, but avoids mixing interface metaphors by eliminating all of the usual buttons of a browser. The interface thus immerses the learner in an environment uniquely associated with the material. Another difference between *ProgramLive* and CIMEL is in their approach to feedback. CIMEL provides an explanation of each of the wrong answers as the student makes these mistakes rather than just providing the correct answer. This helps the student to gain a better understanding of the thought process involved in solving problems.

An ongoing project at Massey University [3] is developing and evaluating an integrated system for web-based education. This system uses web-based delivery of course material including interactive multimedia presentations, problem solving and simulation environments in which students learn by doing. Like CIMEL, TILE (Technology Integrated Learning Environment) provides students with an interactive multimedia environment, and developers with a framework for managing, authoring, monitoring and evaluating multimedia. The most salient differences are that the CIMEL framework lets students go beyond the lessons through collaboration with experts (e.g., instructors, TAs, research librarians and other student) as well as through tools that allow the student to explore current research trends in course-related literature.

The Interactive Learning Modules (ILM) presents web-based multimedia tutorials, created with the Director authoring environment [4]. ILM provides a mechanism for the creation of supplementary material for lectures, and collaborative problem solving environments. The system is highly modular to encourage the usage of parts of the lesson material in different courses. Similarly, the CIMEL multimedia framework is being developed with modularity in mind, where each screen is a separate Flash movie, and screens are organized hierarchically into sections and chapters. (We chose Flash instead of Director or Authorware because its vector-based graphics lets us easily use the whole screen, regardless of resolution.) The CIMEL dynamic tracks interface will let instructors and students create and traverse their own learning track, corresponding to their unique requirements.

1. Glenn D Blank, William M. Pottenger, G. Drew Kessler, Soma Roy, David R. Gevry, Jeffery J. Heigl, Shreeram Sahasrabudhe and Q. Wang. Design and Evaluation of Multimedia to Teach Java and Object Oriented Software Engineering. *American Society for Engineering Education*. June, 2002.
2. D. Gries and P. Gries. *ProgramLive - Master Java Programming in a Self-paced Learning Environment.* New York, John Wiley and Sons, Inc., 2002.
3. C. Jesshope, E. Heinrich and Kinshuk. *On-line Education using Technology Integrated Learning Environments.* Massey University, New Zealand. www-tile.massey.ac.nz/publicns.html.
4. D. M. Millard, Interactive Learning Modules for Electrical Engineering Education and Training. In *Proceedings of the American Society for Engineering Education*, 1999.

Author Biographical Sketches

GLENN D. BLANK

Dr. Glenn D. Blank (glenn.blank@lehigh.edu) is an Associate Professor of Computer Science and Engineering at Lehigh University. His work as principal investigator for the CIMEL project builds on his experience as the lead author, designer and developer for *The Universal Machine: A Multimedia Introduction to Computing* (McGraw-Hill, 1998).

WILLIAM M. POTTENGER

Dr. William M. Pottenger (pottenger@cse.lehigh.edu) is P.C. Rossin Assistant Professor of Computer Science and Engineering at Lehigh University. His research as a co-principal investigator for the CIMEL project builds on his existing research program in textual data mining.

JEFFREY J. HEIGL

Jeffrey Heigl (jjh5@lehigh.edu) is a graduate student at Lehigh University in Bethlehem, Pennsylvania pursuing a M.S. degree in Computer Engineering. He received his B.S. degree in Computer Engineering at Lehigh University in 2001. Jeffrey is a research assistant working on the CIMEL Multimedia Team.

SOMA ROY

Soma Roy (sor4@lehigh.edu) is currently a Research Assistant in the Textual Data Mining Lab in the Department of Computer Science and Engineering at Lehigh University. Her work involves the development of the methodology and multimedia interface for detection of incipient emerging trends and development of constructive exercises for the ADT module.

SHREERAM A. SAHASRABUDHE

Shreeram A. Sahasrabudhe (sas4@lehigh.edu) received his B.E. in Computer Technology from Nagpur University, Nagpur, India in 2001. He is a currently pursuing an M.S. degree in Computer Engineering at Lehigh University. He is a Research Assistant working under Professor Glenn D. Blank on the CIMEL Multimedia Team.