Computer Science Apprenticeship: Creating Support for Intermediate Computer Science Students

Amnon Shabo

Mark Guzdial

John Stasko

College of Computing
Georgia Institute of Technology
Atlanta, GA 30332-0280
USA
{amnonh, guzdial, stasko}@cc.gatech.edu

Abstract: In recent studies of students in intermediate computer science courses, we have noted a number of misconceptions and other learning problems. Supporting intermediate computer science students has not been explored in the literature as well as supporting novice and expert programmers. We are developing an approach focussed on providing a cognitive apprenticeship in computer science. Our approach has four technological components: interactive notes module, a collaborative tool, a case library and problem-solving activities accompanied with various scaffolding features including annotated links to the other modules and drawing upon existing resources. Our first focus is on undergraduate Computer Graphics where we are beginning to evaluate our materials.

I. Addressing the Problems of Intermediate Computer Science Students

In recent studies of students in intermediate Computer Science courses (e.g., Computer Graphics, object-oriented programming), we have noted a number of misconceptions and other learning problems (e.g., a tendency to create too-centralized object-oriented programs [Guzdial, 1995a]). This is an area not as well explored as support for Computer Science beginners and experts. Solutions that have had success in supporting novice students include intelligent tutoring support [Anderson, Corbett, Koedinger, & Pelletier, 1995], learning environments based on structured editors [Miller, Pane, Meter, & Vorthmann, 1995], and learning environments based on high-level cognitive structures with supportive component libraries [Soloway, Guzdial, Brade, Hohmann, Tabak, Weingrad, et al., 1993]. Technologies that have had success with experts include intelligent programming support such as the Programmer's Apprentice [Waters, 1981] and reusable component libraries [Guindon & Curtis, 1988].

Intermediate students do not have the knowledge necessary to take advantage of expert-level tools, yet they are also already beyond many of the supports that have been successfully used with novices, e.g., structured editors are less useful with students who have already been able to learn the syntax and semantics of two or more programming languages. Two commonalities between the sets of approaches are:

- Intelligent support for diagnosing and addressing problems.
- Libraries of examples and components.

One of the particular challenges of working with intermediate students is that they are already taking on problems that are broader and deeper than those that novices take on. Support for their learning problems must reflect that breadth and depth. The intelligent support must draw upon a large knowledgebase, and the libraries must be broad enough to encompass the kinds of needs that intermediate students will bring to it.

Fundamentally, the approach that we are taking is to conceive of intermediate Computer Science as a kind of cognitive apprenticeship [Collins et al., 1989]. We want to provide students with a broad and deep library of resources, with intelligent support for identifying and using relevant resources. We also want to create learning communities where students can help one another and more expert students can help other students. Our approach centers around four pieces:

- A collection of resources which we broadly call Interactive Notes. For each intermediate course, there are a collection of resources already existing: course notes, glossaries, quizzes, visualizations, simulations, algorithm animations [Stasko, Badre, & Lewis, 1993], etc. We are collecting these and integrating them to provide breadth and depth beyond the cases themselves [Carlson, Guzdial, Kehoe, Shah, & Stasko, 1996].
- A computer-supported collaborative learning tool, CaMILE [Guzdial et al., 1996], [Guzdial, Turns, Rappin, & Carlson, 1995], which encourages collaboration with links to external resources.
- A case library [Kolodner, 1993] of relevant stories and problems, told in a simplified form but within the context of real-world expert stories.
- Problem-Solving Activities supported by various scaffolding features such as annotated links to the case library and relevant notes pages. Scaffolding for accessing and using the resources is fadeable to be adaptable to the needs of individual students [Guzdial, 1995b].

All three of our tools are based on the World Wide Web, which eases usability and access problems. Our first efforts are aiming at the undergraduate Computer Graphics course at Georgia Tech, and we are beginning an effort in the introduction to object-oriented programming course as well.

II. The Interactive Notes Module

The interactive notes module is in advanced stage of progress. Notes of the introductory computer graphics course given at the College of Computing at Georgia Tech are available in the project Web site, linked with interactive elaboration pages for some of the topics (e.g., visualization tools for exploring two-dimensional transformations). We work closely with instructors of this course in trying to gather two types of data: (a) major difficulties and misconceptions students have in learning computer graphics and (b) usability and attitudes during use. The problems and misconceptions analyses are based mainly on corrected students assignments such as homework, programs, quizzes, and finals.

In a survey conducted among students of the introductory computer graphics course during its last offering, more than 70% agreed that the pages of the interactive notes module were good and useful overall. However, only 20% agreed that the pages helped them to solve assignments in the course, and a relatively low 60% agreed that the pages helped them to study for the course. These results strengthen our belief that simply providing resources is not enough to help students in using the knowledge from the course in their work, and encourage us to develop the other modules in the system and link them together to provide a more effective learning environment.

III. CaMILE: Collaborative and Multimedia Interactive Learning Environment

CaMILE is a Web-based collaboration tool which has been used in several courses outside of Computer Science [Guzdial, et al., 1996]. CaMILE provides a facility for students to discuss issues and share media [see Fig. 1]. What makes CaMILE more supportive of learning than newsgroup or electronic mail include:

- CaMILE uses procedural facilitation (as in the CSILE system [Scardamalia, Bereiter, McLean, Swallow, & Woodruff, 1989]) to encourage reflection on the roles of the notes in the collaboration.
- Threads of discussions (a collection of related notes) can be directly addressed as a kind of sub-discussion or contextualized discussion. Using this feature, an instructor can create a discussion on a particular topic (e.g., a new assignment) and create a link to that discussion, so that students using that link need not ever see the entire discussion and can instead focus on the relevant parts of the discussion.
- Notes in CaMILE can be linked to any kind of media, both based on the Web and based on the student's personal computer (e.g., spreadsheets, drawings, word-processing documents). This linking facilitates integrating diverse media and encourages students to contribute their own work in the collaboration.

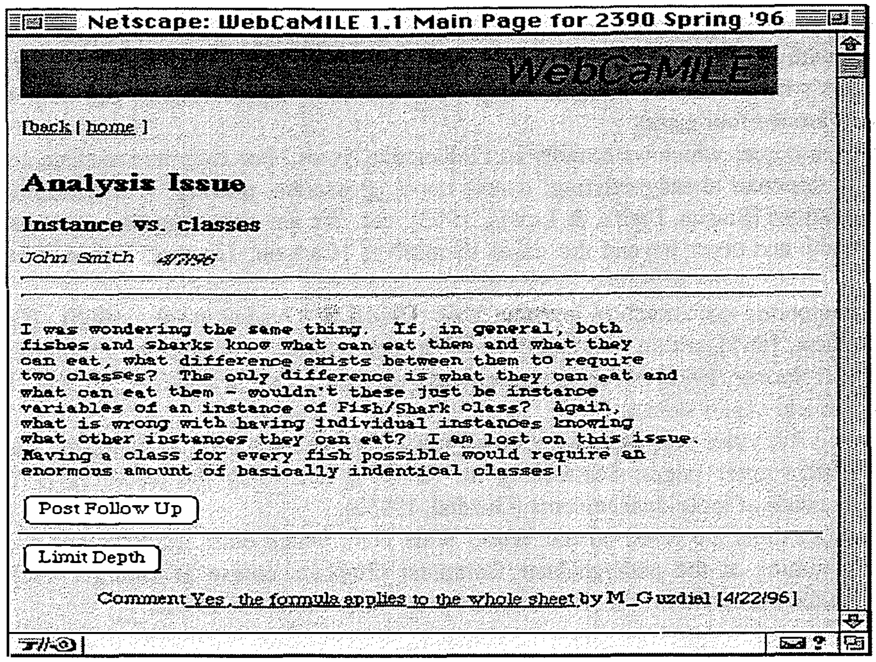

Figure 1: Sample note from WebCaMILE.

IV. The Computer Graphics Case Library

A prototype of a computer graphics case library has been completed. A case library is typically aimed at storing previous cases in a domain for the sake of finding solutions to problems encountered in new cases. A goal for this case library is to enable broad use. While we hope that experts might use it to find cases similar to problems in which they are currently engaged, the central focus is for intermediate students to use the library to learn about major problems in computer graphics.

To broaden the stories in the case library, we use resources at the Graphics, Visualization and Usability (GVU) Center at Georgia Tech and resources in the entire computer graphics community and invite users to add new stories and lessons (i.e., lessons drawn out of the stories.) The contributions would be moderated by experts and educators. Thus, the library is likely to be a live and growing casebase. Our hope is that the library might become large enough to be useful for experts. If we are successful, the use of the library and contribution of new cases could ensure the real-world characteristic of this learning environment for learners. In addition, educators' contribution of simplified stories and learning problems could serve as the library internal bridge between students and experts.

Each story in the case library may consist of important milestones in the development, intermediate products, source code segments, visualization of different stages, tools, methods, etc. In a concurrent process we are trying to draw out lessons from the incoming stories in a form of a common problem accompanied by different solving strategies. Thus, each common problem is linked to variety of stories that address this problem. The incoming problems, issues, solutions and stories are then indexed to cover all major concepts and terms in the domain.

An example in our prototype is the story of a flight simulator aimed at people who suffer from fear of flying [Hodges, Rothbaum, Watson, Kessler, & Opdyke, 1996]. The project is complex and entails the use of many algorithms and methods in Computer Graphics. One of the general issues related to this story is Hidden Surface Elimination and different solving strategies are Z-buffering, BSP Tree, Back-Face Removal, Depth-Sort and the Scanline Algorithm. Part of the data described in the story is as follows: *"...our model contains a detailed set of terminal buildings occupying one square mile, as well as a crude representation of the terrain within a 50 mile radius. The model as a whole contains 100,000 polygons, with over 90,000 of them used to describe the terminal buildings."* One of the problems told in this story is the phenomena of 'Bleeding Polygons' which is a result of using the Z-buffering method in such a model. A learning problem (within the problem-solving activities) associated with this story is: *"What sort of inadequacies with Z-buffering might this sort of model reveal?"*

In general, a specific lesson is a common problem (or a general issue) along with its approaches, solving strategies or solutions. When multiple solutions exist they are presented in a table [see Fig. 2] to facilitate the comparison of the pros and cons of each solution. Thus, while each story might have its own structure, lessons are presented in a more structured way.

A searching mechanism with partial matching [Kolodner, 1993] is available for the user to search the library. Concept maps in the interactive notes and navigational maps in the case library could serve as alternatives for formulating a logical query with index terms. Also, these maps might help in getting an overview of the system content and structure.

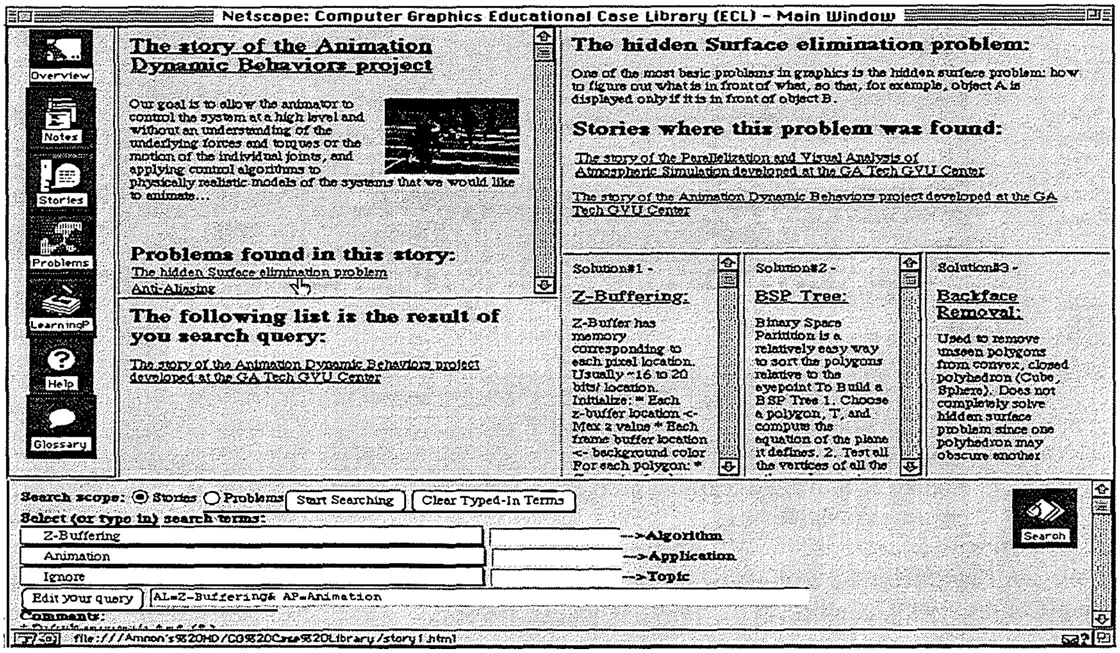

Figure 2: An example of a query, identified in the bottom frame. Clicking on the retrieved story brings its review (top-left frame), and clicking on the related problem of Hidden Surface Elimination brings on the right the lesson: an abstracted problem and its different solving strategies.

V. Problem-Solving Activities and Scaffolding

The educational use of the system can be done in several ways such as searching the case library or browsing the interactive notes pages to learn about main issues and problems in computer graphics. However, a more structured way of learning with the system is through the problem-solving activities. In this module, the learner is able to select a learning problem from a repository of learning problems. Some of these problems are derived form the case library stories. Thus, it is possible to challenge the learner to look for different solutions by retrieving relevant stories from the library. Part of the scaffolding mechanism is to provide at this point annotated links to those relevant stories, links which might ease the finding and reading of significant portions in each story. The same concept is being implemented by providing links to the relevant pages from the interactive notes module which, from the point of view of the problem-solving process, can be seen as a resource of factual knowledge. All these links are meant to support the learner in his/her effort to solve the selected learning problem.

Since lessons were drawn out of real-world stories, the learner has the opportunity to select a problem that fits with his/her interest or need resulting in possible increase of motivation to learn [Schank, 1992]. The way a learning problem is presented to the learner [see Fig. 3] has some similarities to the problem-based learning approach [Barrows, 1986]. The first two pieces are the facts and learning issues need to be explored. The learner is able to type-in learning issues or get suggestions from the system by accessing one of the scaffolding features [see Fig. 4]. Learners can then request to access peer review . These peer solutions are based on authentic answers taken from finals or other assessments in the computer graphics course. The answers were analyzed and selected as representatives of common problems and possibly misconceptions in understanding the issue [see Fig. 5].

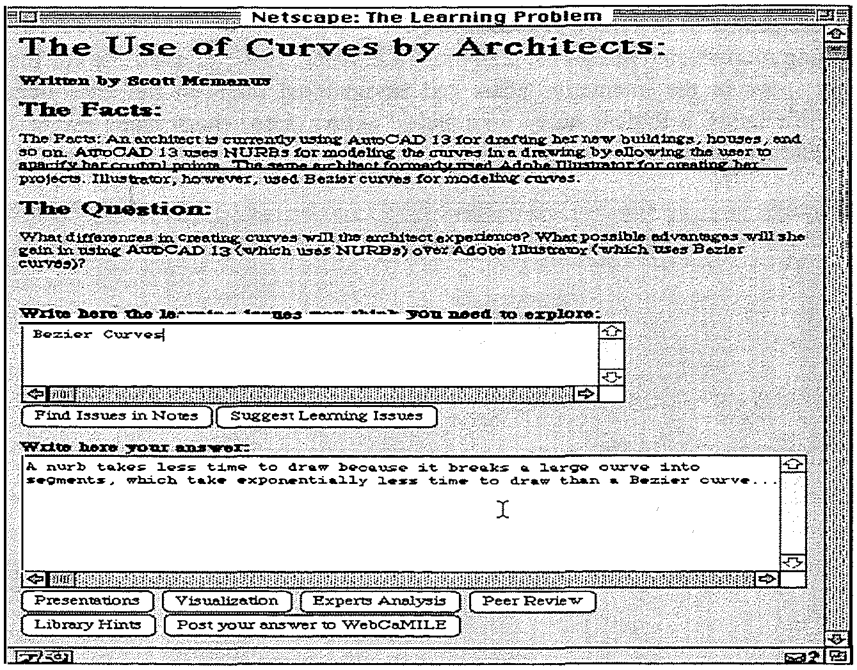

Figure 3: Solving a learning problem selected from a list of questions presented to the learner.

We can characterize the scaffolding in the system as falling into three categories of software-realized scaffolding [Collins, 1990], [Collins, et al., 1989], [Guzdial, 1995b]:

- Modeling or Communicating Process where students are helped with understanding the problem and what an expert's process might be. Examples of this kind of scaffolding features include: [see Fig. 4]
  - learning issues: links to notes pages where relevant learning issues are discussed. From these pages, elaboration is available accompanied with many examples;
  - library hints: annotated links to story portions of greater importance to solving the selected problem;
  - experts analysis: analysis of the problem is provided by at least two experts, to emphasize that there is usually more than one 'correct answer' ;
  - presentations: images, animations, movies and other kinds of presentations that support the understanding of the selected problem;
- Coaching where students are provided help which responds to their given selections and responses. Examples include: [see Fig. 5]
  - peer review: showing students incorrect/incomplete answers beside their own answer so that they might be supported in identifying their own misconceptions;
  - visualization: tools that can aid in understanding basic issues such as a 2D visualization tool or a curve construction tool. Some of these tools might have their own unique interaction providing the learner with detailed feedback for his/her doing [Friedler & Shabo, 1989/90].
- Articulating where students are provided with opportunities to express their solutions and evaluations. Also, the articulation could lead to collaboration and the system facilitates this by enabling the students to post their articulations to relevant WebCAMILE discussion [see Fig. 3].

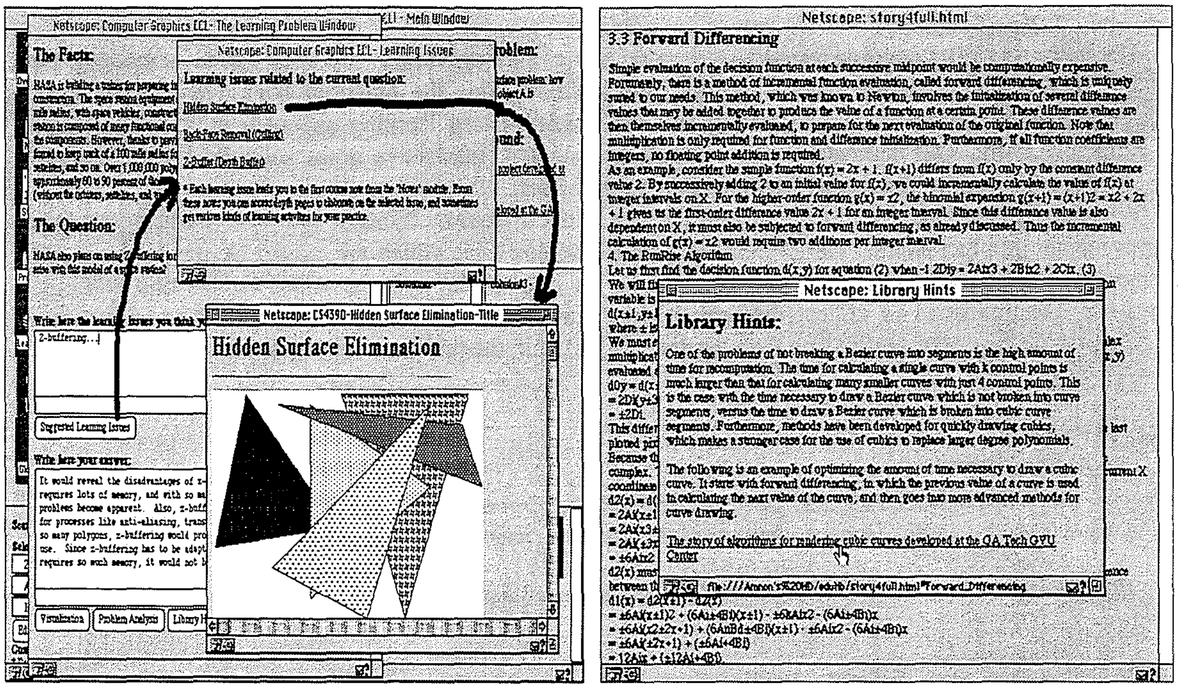

Figure 4: *Communicating the process-* On the left- an example of suggested learning issues (in the small window on the right side) where clicking on any issue brings up the corresponding notes/depth page. On the right- an example of the 'library hints' scaffolding feature, where a link to a section 3.3. in a story about cubic curves is annotated to ease the reading of this section.

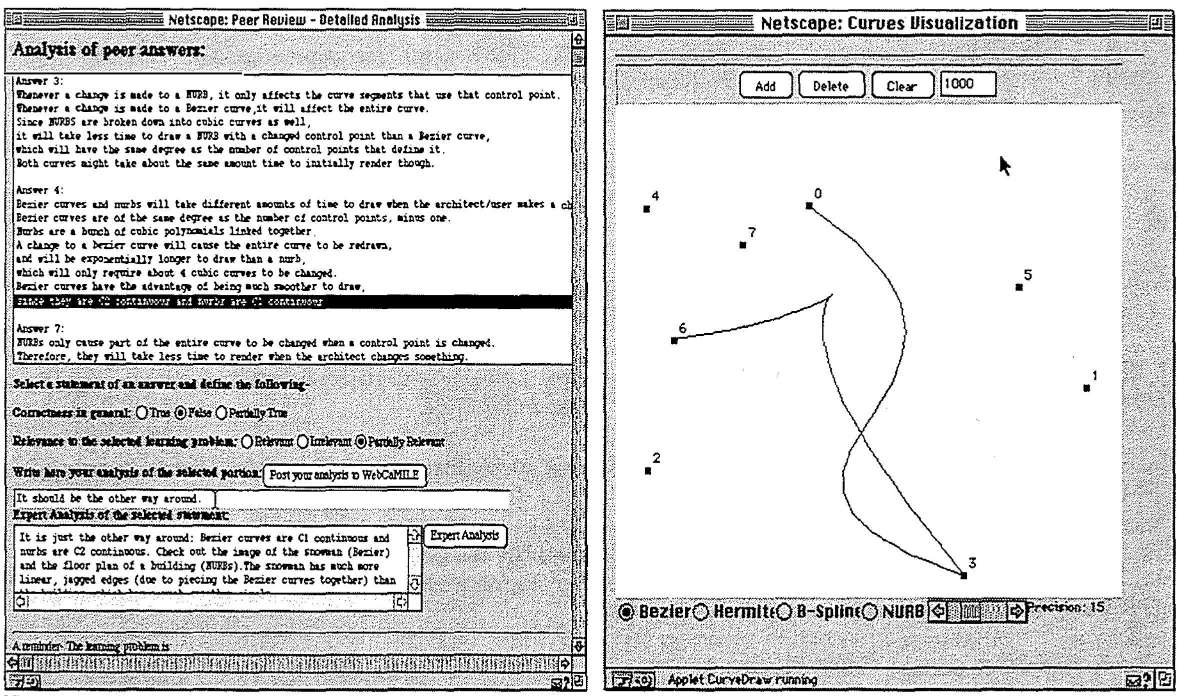

Figure 5: *Coaching-* On the left- an example of detailed analysis of peer answers for the selected learning problem where each statement in an answer has an expert analysis available for it but no feedback is given for the overall answer. On the right- a visualization tool[1] for curves where the learner is able to construct various types of curves based on the same control points.

[1] Technical note: this tool was developed as a Java Applet while most of the other parts of the system are written in JavaScript embedded in HTML files.

VI. Summary

The paper describes ongoing research to develop a cognitive apprenticeship approach to addressing the needs of the intermediate computer science student. We are examining the potential cognitive benefits from integrating the apprenticeship model of learning and the problem-based learning approach in a computerized framework on the Web. Our approach involves four components: interactive notes module, a collaborative learning environment, a case library and problem-solving activities.

We are currently engaged in formative evaluation of the system by trying it out with students from an advanced computer graphics course. The evaluation is based on log files which are created by the software and contain most of the actions the student has done during the use of the system. Also we use observations, interviews and questionnaires to collect data on student's attitudes towards the system and to get deeper understanding on the ways the scaffolding features are being used in the problem-solving process.

VII. References

[Anderson, Corbett, Koedinger, & Pelletier, 1995] Anderson, J. R., Corbett, A. T., Koedinger, K. R., & Pelletier, R. (1995). Cognitive Tutors: Lessons Learned. *Journal of the Learning Sciences*, *4*(2), 167-208.

[Barrows, 1986] Barrows, H. S. (1986). A taxonomy of problem-based learning methods. *Medical Education*, *20*, 481-486.

[Carlson, Guzdial, Kehoe, Shah, & Stasko, 1996] Carlson, D., Guzdial, M., Kehoe, C., Shah, V., & Stasko, J. (1996). WWW Interactive Learning Environments for Computer Science Education. In *ACM SIGCSE'96 Technical Symposium*, (pp. 290-294). Philadelphia, PA: ACM.

[Collins, 1990] Collins, A. (1990). Cognitive apprenticeship and instructional technology. In B. F. Jones & L. Idol (Eds.), *Dimensions of Thinking and Cognitive Instruction* (pp. 121-138). Hillsdale, NJ: Erlbaum and Associates.

[Collins et al., 1989] Collins, A., Brown, J. S., & Newman, S. E. (1989). Cognitive apprenticeship: Teaching the craft of reading, writing, and mathematics. In L. B. Resnick (Eds.), *Knowing, Learning, and Instruction: Essays in Honor of Robert Glaser* (pp. 453-494). Hillsdale, NJ: Lawrence Erlbaum and Associates.

[Friedler & Shabo, 1989/90] Friedler, Y., & Shabo, A. (1989/90). Developing a high level database to teach reproductive endocrinology using the Hypercard program. *Computers in Mathematics and Science Teaching*, *9*(2), 55-66.

[Guindon & Curtis, 1988] Guindon, R., & Curtis, B. (1988). Control of cognitive processes during software design: What tools are needed? In *CHI'88: Conference Proceedings: Special Issue of the ACM/SIGCHI Bulletin* (pp. 263-268).

[Guzdial, 1995a] Guzdial, M. (1995a). Centralized mindset: A student problem with object-oriented programming. In *ACM SIGCSE Technical Symposium 1995* (pp. 182-185). New York: ACM Press.

[Guzdial, 1995b] Guzdial, M. (1995b). Software-realized scaffolding to facilitate programming for science learning. *Interactive Learning Environments*, *4*(1), 1-44.

[Guzdial et al., 1996] Guzdial, M., Kolodner, J. L., Hmelo, C., Narayanan, H., Carlson, D., Rappin, N., Hübscher, R., Turns, J., & Newstetter, W. (1996). Computer support for learning through complex problem-solving. *Communications of the ACM, In press.*

[Guzdial, Turns, Rappin, & Carlson, 1995] Guzdial, M., Turns, J., Rappin, N., & Carlson, D. (1995). Collaborative support for learning in complex domains. In J. L. Schnase & E. L. Cunnius (Eds.), *Computer Support for Collaborative Learning (CSCL '95)* (pp. 157-160). Bloomington, IN: Lawrence Erlbaum Associates.

[Hodges, Rothbaum, Watson, Kessler, & Opdyke, 1996] Hodges, L. F., Rothbaum, B. O., Watson, B. A., Kessler, G. D., & Opdyke, D. (1996). A virtual airplane for fear of flying therapy. In *Proceedings of VRAIS '96* IEEE Virtual Reality Annual Symposium.

[Kolodner, 1993] Kolodner, J. (1993). *Case Based Reasoning*. San Mateo, CA: Morgan Kaufmann Publishers.

[Miller, Pane, Meter, & Vorthmann, 1995] Miller, P., Pane, J., Meter, G., & Vorthmann, S. (1995). The Genies of Carnegie Mellon University: Programming environments we have known and loved. *Interactive Learning Environments*, To appear.

[Scardamalia, Bereiter, McLean, Swallow, & Woodruff, 1989] Scardamalia, M., Bereiter, C., McLean, R., Swallow, J., & Woodruff, E. (1989). Computer-supported intentional learning environments. *Journal of Educational Computing Research*, *5*(1), 51-68.

[Schank, 1992] Schank, R. C. (1992). *Goal-based scenarios* (Technical Report No. The Institute for the Learning Science, Northwestern University.

[Soloway, Guzdial, Brade, Hohmann, Tabak, Weingrad, et al., 1993] Soloway, E., Guzdial, M., Brade, K., Hohmann, L., Tabak, I., Weingrad, P., & Blumenfeld, P. (1993). Technological support for the learning and doing of design. In M. Jones & P. H. Winne (Eds.), *Foundations and frontiers of adaptive learning environments* (pp. 173-200). New York: Springer-Verlag.

[Stasko, Badre, & Lewis, 1993] Stasko, J. T., Badre, A., & Lewis, C. (1993). Do algorithm animations assist learning? An empirical study and analysis. In *Proceedings of INTERCHI'93* (pp. 61-66). Amsterdam, The Netherlands: ACM.

[Waters, 1981] Waters, R. C. (1981). The Programmer's Apprentice: A sessions with KBEmacs. *IEEE Transactions on Software Engineering*, *11*(11), 1269-1320.

Acknowledgments

This work was supported in part by the EduTech Institute through a grant from the Woodruff Foundation and in part by NSF Grant # CDA 9414227.