How can we cost-efficiently help students use software to construct knowledge?

Supporting Learners as Users

Mark Guzdial
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332
guzdial@cc.gatech.edu

Software designers following a user-centered design philosophy try to provide software that helps users achieve their task goals. Word-processing programs help users to create, format, and spellcheck text, if that's the goal of the users. Checkbook programs help users to record transactions and reconcile their account with the bank, if that's their goal.

But sometimes, the user's goal is not the same as the explicit task that the software was designed for. That's the state that learners are in when they attempt to meet their learning goals through performing some task. For example, you may want to learn about physics by programming physics simulations, learn something about statistics by exploring in a spreadsheet or with Mathematica, or learn about yourself by keeping a personal journal in a word-processing program. Many adults in lifelong learning programs are attempting this kind of learning, where they are using their real work workstations to support their learning activity, too. The learning becomes even more complicated if the learner does not already know how to program or to use spreadsheets or Mathematica. This latter situation is common in undergraduate education where a student is asked to use the tool used by professionals while still learning the content of their domain (e.g., an engineering student learning to use a mathematical modeling language like Matlab while learning mathematical modeling or digital signal processing).

Learners don't have a single goal or task when they are using an application program, but two or more. The goals may be hierarchically arranged in terms of importance, but they are often pursued in parallel. Primarily, their task is to learn some content—or maybe, to get a good grade, and to do whatever learning is necessary to get that good grade. Secondarily, their task may be to program, to build a spreadsheet, or to write. Related to that secondary goal may be to *learn* to program, to build a spreadsheet, or even to use the word processor. I call this model of learner as *learner-as-user*, which contrasts with:

- *Learner-as-learner*, where the one-and-only goal is to learn. Computer-assisted instruction and intelligent tutoring systems support this model of the learner.

Learners-as-users, with a huge hierarchy of goals, need...*scaffolding* to insure that the enormity of lower-level goals does not prevent achieving the uppermost goals.

- *Expert-as-user*, where the expert knows her goal and is using software to achieve a known task.

Supporting learners-as-users is a complicated challenge. I have argued elsewhere that designing for learners is a wholly different activity than traditional user-centered design, which tends to focus on expert users (Soloway, Guzdial, and Hay, 1994). Learners are less motivated (or, perhaps just need more motivation to get through all the learning and doing tasks ahead of them), are changing in their knowledge and task characteristics more rapidly than experts, and know less about what they are doing. Even from a user-centered design perspective, most current applications and support (e.g., documentation, training, etc.) are not aimed to support the learner-as-user, since they rarely include support for the task of learning.

Consider instances of good support for users, such as the "minimal manual" approach (Carroll, Smith-Kerker, Ford, and Mazur, 1986). A minimal manual organizes documentation in terms of the tasks in which users will actually be engaging. Our current best methodology for designing minimal manuals is to use the GOMS model (Card, Moran, and Newell, 1983) to identify the users' tasks and how they should be deconstructed in the manual (Gong and Elkerton, 1990). But GOMS is insufficient for modeling the activities in learners—GOMS assumes that the user knows their goals and operators, not that they will be learning them and having to choose between different goals in a hierarchy (John, 1995). Our current best practice is insufficient for meeting the needs of learners, and new methodologies and perhaps new kinds of support are needed to meet the needs of learners-as-users.

Learning is a conscious task, most cognitive science researchers now agree (Bruer, 1993). Experience in project-based learning suggests that learners *do* need support in understanding how to use existing applications to support their activities (Blumenfeld et al., 1991). Learners-as-users, with a huge hierarchy of goals, need enormous motivation to work through all of them, models of good process and good product to emulate, and scaffolding to insure that the enormity of lower-level goals does not prevent achieving the uppermost goals. Scaffolding is an education term used to describe the kind of support that a master craftsperson might offer to an apprentice in a traditional apprenticeship learning situation (Collins, Brown, and Newman, 1989; Rogoff, 1990). A characteristic of most scaffolding is that it *fades*, or diminishes so as to allow the student to take over the role of the support. However, sometimes you want scaffolding to remain permanently, because fading the scaffolding would introduce a new and unnecessary learning goal (Hmelo and Guzdial, 1996). Scaffolding is one of the kinds of support that is needed to help the learner succeed as a user.

The question is how to provide the kind of support that a learner-as-user needs to succeed at the tree of goals facing her. In the rest of this paper, I present several instances of supporting learner-as-user. The first few are offered as relevant background, to show that learners can be successfully supported as users and to show how that has been achieved in the past. The latter two are instances from my own recent work at Georgia Tech's College of Computing in a class where students are learning object-oriented design and analysis, which are independent of any programming language, while they learn to program in a particular programming language, Smalltalk. In this more recent work, the students are being supported in an unmodified, commercial piece of software in a traditional classroom setting, and thus the work offers lessons on how to provide support to learners-as-users without modifying the software or the learners' situation.

Successful Support for Learner-as-User

Perhaps the first successful research study of learner-as-user was the thesis work of Idit Harel at MIT's Media Lab with Seymour Papert (Harel, 1991; Harel and Papert, 1990). Harel asked fourth graders to build educational software for third graders to help teach about fractions. In the fourth graders's goal hierarchy were the goals of learning the programming language Logo (an unmodified, commercial version), learning to program at all, learning to design, learning what educational software was about, and, most importantly, learning fractions themselves. Harel found (through interviews and standardized tests) that her fourth graders really did not understand fractions when they began her project, nor did they know anything about programming.

By the end of the project, every student had developed a relatively complex piece of interactive software, and had learned a lot about Logo and fractions as well. In fact, the students who were part of Harel's project learned significantly more on all measures than students in control groups who had studied both Logo and fractions in more traditional settings. Harel's work was a great success for those who felt that learners could succeed as users. She showed that the hierarchy of goals could work in synergy to support better than expected learning for all of the goals. Harel's work was replicated and extended by Yasmin Kafai in several studies that explored very interesting variations (Kafai and Harel, 1990; Kafai and Harel, 1991a; Kafai, 1993), such as where the next group of fourth graders was aided by the (then) fifth graders as consultants.

Harel and Kafai supported their learners-as-users through a number of structures:

- **Time.** The implementations of this project had students working daily for four months or more. That is much more time than is typically spent on any single project at any level of education.
- **A great deal of hands-on mediation.** Besides the teacher and Harel or Kafai, additional graduate students were always in the room. In general, for a group of less than 20 students, there was often three teachers available to help. The support of the teachers took various forms, from answering questions, to prompting for reflection, to pointing out other students who might offer help.
- **Collaboration.** Students worked on their own projects, for the most part. There were no group efforts. Nevertheless, students were all working on the same kind of project at the same time, and collaborations naturally sprang up. Kafai calls this "collaboration in the air" where students working on individual but related projects can team up to lend support, thus supporting both the project performance goals and the learning goals (Kafai and Harel, 1991b).
- **A task with intrinsic motivation.** Students in Harel and Kafai's studies were encouraged to pick their own kind of activity within the general goal of helping other students to learn. Students were motivated because the project was their own, involving media of interest, and with an interesting goal.
- **Specific support on-demand.** While the students were mostly left to work during their project time, they could request specific lectures such as on how to do graphics in Logo or how to design a screen. When requested, a lecture on the subject would be provided to the students.
- **Reflection for learning.** Both Harel and Kafai kept students to a daily ritual of writing in their Journal to capture the day's progress and of writing up their Plans for the next day. Reflection and review is a critical component of learning (Collins and Brown, 1988). That alone probably explains much of the learning success of the projects.

The work of Harel and Kafai has served as a model for supporting learners-as-users in other research projects (e.g., (Lehrer, 1992)), but it's an expensive model. It requires a great deal of labor and time. Cheaper and faster forms of support for learners-as-users have typically involved the use of specialized software. This "specialized software" is still software supporting a useful task other than learning (such as programming, data analysis, and visualization), but is especially designed to support learning, too. In contrast, most educational software aims at sup-

porting learning (e.g., CAI, tutors, computer games), but does not also support the performance of a useful task other than learning.

The Boxer project at the University of California at Berkeley, led by Andrea DiSessa, has successfully supported student learning in math (Adams and DiSessa, 1991), physics (Sherin, DiSessa, and Hammer, 1992), biology (Ploger, 1991), and other fields while the students learned to program in a specialized programming language called Boxer (DiSessa and Abelson, 1986; DiSessa, Abelson, and Ploger, 1991). Boxer is similar to the Logo programming language used by Harel and Kafai, but it offers special user interface additions which make it particularly well suited for learning programming and for using Boxer as an environment for working in, not just programming in. For example, all computation in Boxer is based on "boxes" which are general interface objects in which text, data, or even graphs can be stored and manipulated. There is little explicit support in Boxer for the elements in Harel's and Kafai's studies, so much of the labor and time intensive elements must still be provided in the classroom.

In my own dissertation work, I attempted to address some of the more expensive time elements. In Emile, I provided explicit and adaptable scaffolding to help high schools program physics simulations as a way to learn physics (Guzdial, 1995; Guzdial, 1993). Emile was similar to the Training Wheels approach (Carroll and Carrithers, 1984a; Carroll and Carrithers, 1984b), in that it prevented common learner error states by simply disallowing them. In Carroll et al.'s work, they determined the most common learner error states in a word processor, then created a new version of the word processor where those states were blocked. Students were able to learn more quickly and were able to transfer their learning to the full word processor.

In Emile, students began with a number of scaffolding "blocks" in place. For example, they could not type any programming code directly (thus preventing syntax errors). Instead, they could only compose code components from a library and customize them in pre-defined ways. Emile extended the Training Wheels approach in that a variety of kinds of blocks were available, and all were adaptable from within the program. When a student felt ready, she could turn off the block that prevented typing code in order to create code elements that could be combined with library elements. Emile was successful in that students using it were able to learn physics and programming, creating four physics simulations in three weeks.

Not all of the specialized software for learners-as-users is just to teach programming. For example, the Investigator's Workshop (IWshop), developed by Elliot Soloway and his Hi-CE Research Group at the University of Michigan, is aimed at supporting learners engaged in scientific inquiry. IWshop helps students to gather data, visualize their data, create models of their data, and compare and contrast their data and models. IWshop explicitly includes support for reflection (e.g., asking students their plans) as well as lots of guidance like that provided by good mediation. It is used in project-based learning classrooms that offer similar mediation and similar models of collaboration to that used in the Harel and Kafai studies (Krajcik, Blumenfeld, Marx, and Soloway, 1994).

While it is good to know that well-designed software can be built to support learners-as-users, most software is aimed at the more traditional computer user who only wishes to achieve their task goals. Thus, most software that a learner-as-user might want to use in combining a task with learning does not have any special features or special support infrastructure to facilitate the successful completion of a hierarchy of goals. The challenge, then, is figuring out how to support students in using traditional software for more-than-traditional activities. In the best case, the support should not require the learners to leave the situations in which they already find themselves, such as the workplace or the university classroom setting.

Supporting Use of Traditional Software in Traditional Settings for Non-Traditional Users

I teach a class at Georgia Tech, *CS2390 Modeling and Design,* in which students are expected to learn object-oriented analysis, design, and programming in a language-independent manner while building simulations and user interfaces in a specific object-

oriented programming language, Smalltalk. Smalltalk is an integrated language compiler and integrated development environment, and it's not an easy thing to learn (see (Carroll, Singer, Bellamy, and Alpert, 1990) for more on the challenge of learning Smalltalk). My students are in the position of being users of unmodified software like that used by professionals in their field, while they are still learning the knowledge of professionals in the field. As a researcher in educational technology, I have taken this opportunity to use my own class in exploring this problem.

The class is fairly traditional. It has been taught for ten-week terms for the last five years, with 75-100 students per class. Typically, there are three to four teaching assistants helping the professor with grading and addressing students' questions. Assignments are chosen to be interesting, but with such a large class, it's hard to believe that they are intrinsically motivating for all. Assignment examples are designing and building a simulation of a subway system, or a presentation device for a multimedia slide show.

In this section, I describe briefly two very different attempts to support learners-as-users in the context of the *Modeling and Design* class. Both attempts center on the mediation and collaboration aspects of the support that Harel and Kafai provided. In the first, students were provided with a static and carefully designed case base of successful projects developed by prior students in the class. In the second, students are, instead, provided with a flexible, collaborative space in which they can provide their own case base for each other.

A Case Base of Projects

STABLE is the SmallTalk Apprenticeship-Based Learning Environment (Guzdial and Kehoe, 1998). It is a case library of some 13 projects, most by students, that are presented across some 1200 web pages. Each of the projects in STABLE is presented in great detail (Figure 1):

- The steps of the entire process (analysis, design, and programming) are decomposed in a hierarchy. Each step is presented at multiple levels of detail, as an attempt at providing learner-specific scaffolding. The first page that a student sees for a given project presents just a schematic view of what happens in that step. The student can move on to a paragraph of detail, to an outline of the activity in the step, and finally to all the results of the step, including analysis representations and any programming code. Thus, the student could *fade in* the support to the level that he needed.
- Multiple representations are provided of the project: of the process, of the product, of interactions between the code elements. Many of these are the students' own work, simply scanned into the system. All the representations are interactive in the sense that clicking on any element leads to other pages about that element.
- Projects are linked to relevant additional information at both the level of the application software (e.g., in the example in Figure 1 "What the middle mouse button is for") and at the level of language-independent analysis and design ideas (e.g., "What is a part-whole relationship?")

STABLE was provided to *Modeling and Design* students to use as they developed their own projects. Students were encouraged to use these as models of good projects (both process and product) in their assignments during the class, and to feel free to reuse any analyses or programming code that they found valuable in STABLE. The assignments were tuned to take advantage of STABLE, but were not significantly different from other assignments made in the same class (Guzdial and Kehoe, 1998) (Guzdial, 1997). My hope was that STABLE would provide some of the elements that led to successful learners-as-users experiences:

- **Mediation** and **Specific support on-demand.** STABLE provided a huge amount of relevant information to support many of the students' goals. Thus, some of the student needs that were met by mediation and on-demand lectures might be met by STABLE.
- **Collaboration.** STABLE encouraged a sense of collaboration, that past students had been this way and that they were providing help to new students. Current students were provided the opportunity to write up their own cases for fu-

ture students.

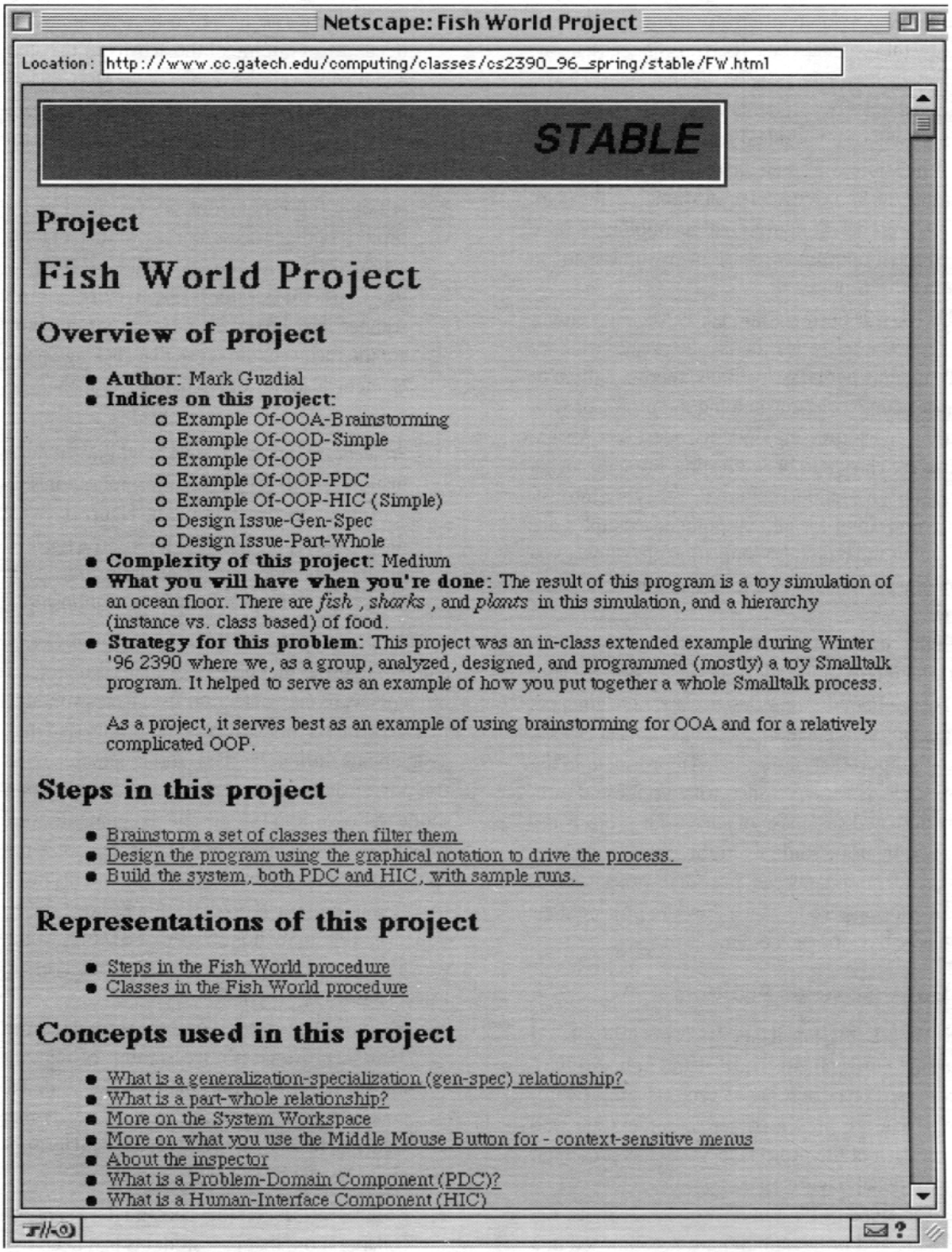

Figure 1: Starting Page for a Project in STABLE

Graduate research assistants and I conducted several studies of STABLE during its first use in *Modeling and Design*. Student performance on projects and exams were compared with prior offerings of the class on similar projects and similar exam questions. We found evidence that STABLE supported students in performance and in learning (Guzdial and Kehoe, 1998).

- **Performance.** Students using STABLE were able to complete a more complicated variation of a problem given to an earlier iteration of the class. On the same grading scale, STABLE students did significantly better, which meant that they were able to use Smalltalk to create a large program successfully.
- **Learning.** Students using STABLE did better on a language-independent design problem on the final exam than did an earlier class on a similar problem.

These two results are significant because they demonstrate support for learners-as-users, without additional resources or time during the class. Even the development cost for STABLE was relatively low since the majority of the cases were based on prior student work, which probably led to improved accessibility for the students than if the cases were drawn from industry or created especially for STABLE. The STABLE results show that a collection of accessible examples with connections to concepts does provide a large piece of the support that learners-as-users need.

However, the STABLE results were not all positive. In surveys and interviews, students kept telling us how much they *disliked* STABLE. They found it confusing, hard to navigate, and difficult to use. Nevertheless, they continued to use it a great deal. For example, the *average* length of a session with STABLE visited 30-40 pages.

Continued analysis of student usage patterns and interviews with students eventually led to the discovery that students use of STABLE was very different than what we expected. Where we provided a great deal of detail within a single project, we found that most students' visits involved *multiple* projects. What students found most valuable about STABLE was comparing and contrasting different projects. They wanted to see how the analysis process on these two projects compared, or how these two different user interfaces were implemented, or how these two different simulations were implemented. The next step is not obvious. Do we create static links between any two pieces of any projects that might be worth comparing? Do we develop a dynamic linking scheme that determines the best matches for the given point in the curriculum and the students' knowledge? Or, is the activity of trying to find two comparable components of projects an important activity that is leading to some of the performance and learning results?

A Collaborative Website that Contains Projects

In parallel with the work on STABLE, my colleagues and I were exploring collaborative learning environments to support project-based learning (e.g., (Guzdial et al., 1997; Guzdial et al., 1996)). At about the time we discovered the problem with STABLE's structure, we had developed a new kind of collaborative space, called a *CoWeb* (for *Collaborative Website,* see http://pbl.cc.gatech.edu:8080/myswiki.1) (Guzdial, 1998). Based on the WikiWikiWeb by Ward Cunningham (http://c2.com), the CoWeb is a website with the simple rule that any page is editable by anyone and anyone can create new pages in the site. We decided to address our case library construction problem by asking students to build their own case library using the CoWeb.

Figure 2 shows a page in the *Modeling and Design* class CoWeb on the left, and on the right is the page that appears if you were to click on the "Edit this Page." The CoWeb allows the user to simply type text as if the page were an e-mail note (e.g., a blank line between paragraphs gets translated into HTML paragraph markers). HTML can be used as the author wishes. Anything between asterisks is translated into a hypertext link. A phrase between asterisks, e.g., ***Mark's Page***, creates a link to a page with that name, creating one if it doesn't already exist. A reference to a GIF or JPEG image address within asterisks is converted into an inline image. A search function and a list of all pages in the date-stamped order in which they were last edited make it easy to track new information and find old information.

We created a CoWeb for the students in the *Modeling and Design* class (at http://pbl.cc.gatech.edu/cs2390/1.html).

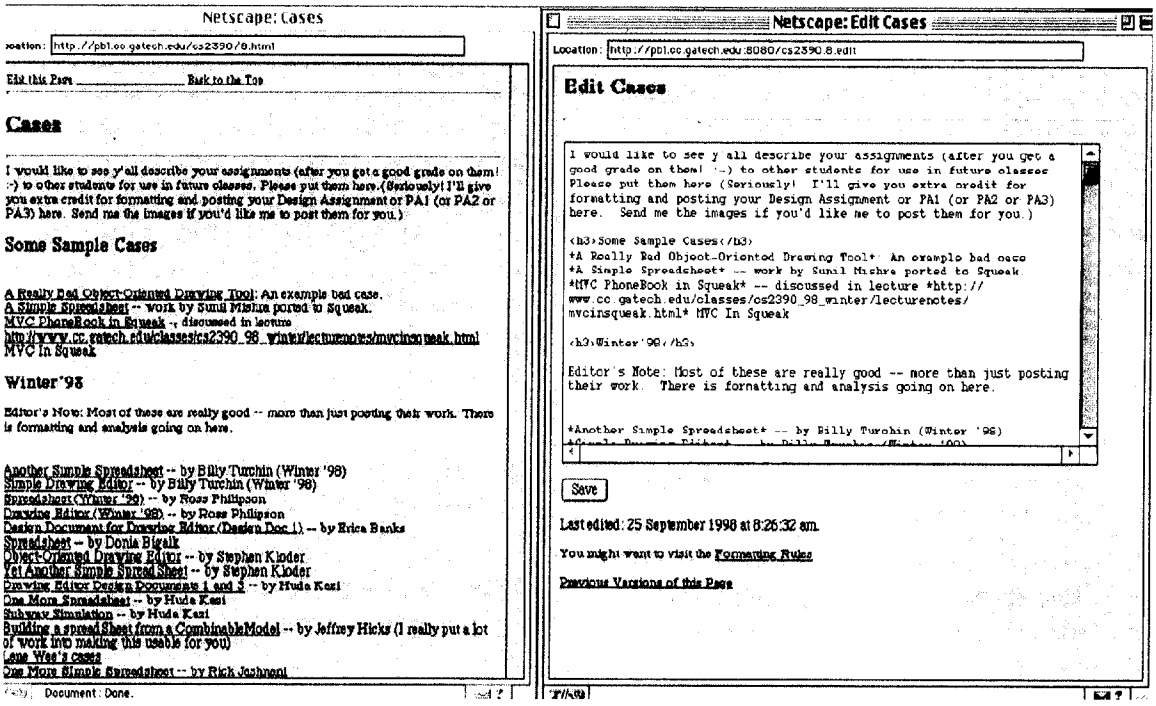

Figure 2: A Page in the Class CoWeb (left) and its Edit View (right).

We created a number of pages for different kinds of discussion and contributions, and created a few initial cases to begin populating a case library page (seen on the left of Figure 2). For example, we created a Smalltalk FAQ page and a page for questions and answers on each assignment. We created a *Who's Who* page where students were encouraged to create their own pages and describe themselves. We offered extra credit to students who would create cases from their best assignments. We also offered extra credit to students who would write essays about issues in the class, from design to programming to Smalltalk.

During the first few weeks, there was relatively little activity in the CoWeb. As the Midterm Exam approached, I posted a set of past midterm exam problems as a review, and created a page for each as a space for posting potential solutions and ask questions. The Midterm Exam Review served as an icebreaker. It was an activity that was valuable to participate in, provided experience in writing in the CoWeb, but was not required. Soon after, all kinds of pages were being developed by the students, from cases (over 20 the first term) to essays on programming in Smalltalk. Non-traditional pages were also created. For example, one student created a page where he and other students could arrange for online gaming competitions.

We have continued to use the same CoWeb in the following offerings of the course. Students are still attempting the same kinds of problems as they were in the STABLE-based offerings of the course. The CoWeb case library is approaching 100 cases. The question-and-answer pages exist in a wide variety of styles. Non-traditional pages also continue to flourish, including an adventure game and even a posting of a song about the class. Valuable new essays and projects have appeared in the CoWeb, from a comparison of Java and Smalltalk, to discussions of design styles—all without my involvement.

We have not measured usage, performance, nor learning as carefully as STABLE yet. However, interviews and surveys suggest that the case library is frequently accessed and is probably as valuable to the students as STABLE was. We believe that the same kinds of mediation and collaboration supports are being provided. Further, the students are not complaining about the structure of the cases there.

What is most striking, however, is the increased motivation that we are seeing in the use of the

CoWeb. We are finding that students who have graduated the class return to answer questions and to update their *Who's Who* pages or cases. Students continue discussions that were started terms ago, sometimes updating or correcting answers that were given in the past. Students spend a great deal of effort constructing pages of valuable information for future students.

The CoWeb, I believe, is providing many of the same supports as STABLE: Good examples linked to conceptual information. But in addition, the CoWeb is providing a collaboration space of which the students have taken active ownership, can contribute where they find most motivating, can find and lend support that best meshes with their goals, and is embedded within a huge support group of peer and more senior students. The CoWeb is motivating to use and update in ways that STABLE never was.

Conclusions

Modern cognitive science sees students not as vessels to be filled, but as active participants in their learning, where the learner is actively constructing knowledge often while constructing something else of value (Papert, 1991). Sometimes, learners use software to help them construct knowledge while constructing other things. In that case, learners are also users.

Students can be successful both as learners and as users who are trying to complete a task, as Harel and Kafai showed, but it can be expensive. We can reduce the costs by providing specialized software and by providing external supports for the learners-as-users. Below is the list, again, of what Harel and Kafai provided. Now, the list contains alternative ways that the support can be provided.

- **Time and a great deal of hands-on mediation** are the two costs that new efforts are trying to reduce. Both STABLE and CoWeb are being used in traditional classes (e.g., no additional teaching assistance, no additional class meetings, the same ten week quarter as before), so the support for learners-as-users is being provided without additional costs.
- **Collaboration.** The work of STABLE and the CoWeb suggests that collaboration can be an important part of support learners-as-users. In some ways, this is not surprising. Amy Bruckman has showed that students can be supports for one another in a MOO (a text-based virtual reality) while they learn the MOO and learn to build and program objects in the MOO (Bruckman, 1994; Bruckman and Resnick, 1995). The CoWeb experience is showing that even traditional students in traditional classes using traditional software can construct the kind of online support systems that enable success for the learner-as-user.
- **A task with intrinsic motivation.** The Boxer and IWshop projects have shown the value of students choosing their own, intrinsically motivating projects and questions, However, the STABLE results in learning and performance were gained even as students tackled traditional, teacher-selected problems. The CoWeb results are showing something different: students can find the activity of supporting one another a motivating one in itself, even if the assignments are fairly traditional.
- **Specific support on-demand.** The case libraries in STABLE and CoWeb meet much of the need for specific support (e.g., questions of how to use the tools), but in both cases, students questions-and-answers are still being addressed via mechanisms such as e-mail and newsgroups, too. Question-and-answer sessions are *always* needed in a learning situation. The issue is how time and labor intensive they are. Good case libraries seem to alleviate some of this cost.
- **Reflection for learning**. IWshop and Emile are our best examples of putting prompts for reflection in the supporting software. Others are also finding ways to prompt students to reflect and learn from their project activities, e.g., (Turns, Newstetter, Allen, and Mistree, 1997).

Supporting learners-as-users is an important challenge that requires new kinds of analysis, development, and implementation mechanisms. The research community is developing good models of what is necessary to support learners-as-users and how to provide this support at a reasonable cost.

An interesting question is whether the learner-as-user model is the exception or the rule. How often

are experts learning new things, or trying to meet multiple learning-and-task goals at once, or finding themselves in situations where they need new expertise? Perhaps the expert-as-user model is the actual exception. If that is so, it may be that more of our support for users at all levels (e.g., professional development and training) needs to attend to features such as collaboration and motivation than it currently does.

Acknowledgements

This research has been funded by the EduTech Institute at Georgia Tech through a grant from the Woodruff Foundation, by the GE Foundation, and by the National Science Foundation (RED-9550458, CDA-9414227). Thanks to colleagues Amy Bruckman, Colleen Kehoe, Janet Kolodner, Mike McCracken, Wendy Newstetter, Jennifer Turns, and others for comments on this paper and more generally in support of this work.

References

Adams, S. T., and DiSessa, A. A. (1991). Learning by "Cheating": Students' inventive ways of using a Boxer motion microworld. *The Journal of Mathematical Behavior, 10*(1), 79-89.

Blumenfeld, P. C., Soloway, E., Marx, R. W., Krajcik, J. S., Guzdial, M., and Palincsar, A. (1991). Motivating project-based learning: Sustaining the doing, supporting the learning. *Educational Psychologist, 26*(3 and 4), 369-398.

Bruckman, A. (1994). Programming for Fun: MUDs as a Context for Collaborative Learning. *Proceedings of the National Educational Computing Conference (NECC'94)* . Eugene, OR: International Society for Technology in Education (ISTE).

Bruckman, A., and Resnick, M. (1995). The MediaMOO Project: Constructionism and Professional Community. *Convergence, 1*(1), 94-109.

Bruer, J. T. (1993). *Schools for Thought: A Science of Learning in the Classroom.* Cambridge, MA: MIT Press.

Card, S. K., Moran, T. P., and Newell, A. (1983). *The Pyschology of Human-Computer Interaction.* Hillsdale, NJ: Lawrence Erlbaum and Associates.

Carroll, J. M., and Carrithers, C. (1984a). Blocking learner error states in a training-wheels system. *Human Factors, 26*(4), 377-389.

Carroll, J. M., and Carrithers, C. (1984b). Training wheels in a user interface. *Communications of the ACM, 27*(8), 800-806.

Carroll, J. M., Singer, J. A., Bellamy, R. K. E., and Alpert, S. R. (1990). A View Matcher for learning Smalltalk. In J. C. Chew and J. Whiteside (Eds.), *Proceedings of CHI'90: Human Factors in Computing Systems* (Seattle, April 1-5, pp. 431-437). New York: ACM Press.

Carroll, J. M., Smith-Kerker, P. L., Ford, J. R., and Mazur, S. A. (1986). *The minimal manual.* Computer Science/Cognition Technical Report: IBM Thomas J. Watson Research Center.

Collins, A., and Brown, J. S. (1988). The computer as a tool for learning through reflection. In H. Mandl and A. Lesgold (Eds.), *Learning Issues for Intelligent Tutoring Systems* (pp. 1-18). New York: Springer.

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

DiSessa, A. A., and Abelson, H. (1986). Boxer: A reconstructible computational medium. *Communications of the ACM, 29*(9), 859-868.

DiSessa, A. A., Abelson, H., and Ploger, D. (1991). An overview of Boxer. *TheJournal of Mathematical Behavior, 10*(1), 3-15.

Gong, R., and Elkerton, J. (1990). Designing minimal documentation using a GOMS approach: A usability evaluation of an engineering approach. *CHI'90 Proceedings* (pp. 99-106). New York: ACM.

Guzdial, M. J. (1993). *Emile: Software-realized scaffolding for science learners programming in mixed media.* Unpublished Ph.D. dissertation, University of Michigan.

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

Guzdial, M. (1997). Technological support for an apprenticeship in object-oriented design and programming. Presented at the OOPSLA'97 Educators Symposium. Atlanta, GA: ACM.

Guzdial, M. (1998). Collaborative websites to support an authoring community on the Web. *Journal of the Learning Sciences,* Submitted.

Guzdial, M., Hmelo, C., Hübscher, R., Nagel, K., Newstetter, W., Puntembakar, S., Shabo, A., Turns, J., and Kolodner, J. L. (1997). Integrating and Guiding Collaboration: Lessons learned in computer-supported collaboration learning research at Georgia Tech. In R. Hall, N. Miyake, and N. Enyedy

(Eds.), *Proceedings of Computer-Supported Collaborative Learning'97* (pp. 91-100). Toronto, Ontario, Canada.

Guzdial, M., and Kehoe, C. (1998). Apprenticeship-based learning environments: A principled approach to providing software-realized scaffolding through hypermedia. *Journal of Interactive Learning Research*, 9, 107-129.

Guzdial, M., Kolodner, J. L., Hmelo, C., Narayanan, H., Carlson, D., Rappin, N., Hübscher, R., Turns, J., and Newstetter, W. (1996). Computer support for learning through complex problem-solving. *Communications of the ACM, 39*(4), 43-45.

Harel, I. (1991). *Children Designers: Interdisciplinary Constructions for Learning and Knowing Mathematics in a Computer-Rich School.* Norwood, NJ: Ablex.

Harel, I., and Papert, S. (1990). Software design as a learning environment. *Interactive Learning Environments, 1*(1), 1-32.

Hmelo, C. E., and Guzdial, M. (1996). *Of Black and Glass Boxes: Scaffolding for Doing and Learning.* Paper presented at the International Conference of the Learning Sciences, Evanston, IL.

John, B. (1995). Why GOMS? *Interactions, 2*(4), 80-89.

Kafai, Y., and Harel, I. (1990). The instructional software design project: Phase II. In I. Harel (Ed.), *Constructionist Learning: A 5th anniversary collection of papers*. Cambridge, MA: MIT Media Laboratory.

Kafai, Y., and Harel, I. (1991a). Children learning through consulting: When mathematical ideas, knowledge of programming and design, and playful discourse are intertwined. In I. Harel and S. Papert (Eds.), *Constructionism* (pp. 110-140). Norwood, NJ: Ablex.

Kafai, Y. B., and Harel, I. (1991b). Learning through design and teaching: Exploring social and collaborative aspects of Constructionism. In I. Harel and S. Papert (Eds.), *Constructionism* (pp. 111-140). Norwood, NJ: Ablex.

Kafai, Y. B. (1993). *Minds in Play: Computer Game Design as a Context for Children's Learning.* Unpublished Ph.D. Dissertation, Graduate School of Education of Harvard University.

Krajcik, J. S., Blumenfeld, P. C., Marx, R. W., and Soloway, E. (1994). A collaborative model for helping teachers learn project-based instruction. *Elementary School Journal, 94*(5), 483-497.

Lehrer, R. (1992). Authors of knowledge: Patterns of hypermedia design. In S. Lajoie and S. Derry (Eds.), *Computers as Cognitive Tools* (pp. 197-227). Hillsdale, NJ: Lawrence Erlbaum Associates.

Papert, S. (1991). Situating constructionism. In I. Harel and S. Papert (Eds.), *Constructionism* (pp. 1-11). Norwood, NJ: Ablex Publishing Company.

Ploger, D. (1991). Learning about the genetic code via programming: Representing the process of translation. *The Journal of Mathematical Behavior, 10*(1), 55-77.

Rogoff, B. (1990). *Apprenticeship in thinking: Cognitive development in social context.* New York: Oxford University Press.

Sherin, B., DiSessa, A. A., and Hammer, D. (1992). Dynaturtle revisited: Learning physics through collaborative design of a computer model. *Interactive Learning Environments, 3*(2), 91-118.

Soloway, E., Guzdial, M., and Hay, K. E. (1994). Learner-centered design: The challenge for HCI in the 21st century. *Interactions, 1*(2), 36-48.

Turns, J. A., Newstetter, W., Allen, J. K., and Mistree, F. (1997). The Reflective Learner: Supporting the Writing of Learning Essays that Support the Learning of Engineering Design through Experience. *Proceedings of the 1997 American Society of Engineering Educators Conference* . Milwaukee, WI.