JavaCAP:
A Collaborative Case Authoring Program on the WWW

Amnon Shabo, Kris Nagel, Mark Guzdial, Janet Kolodner

EduTech Institute, College of Computing, Georgia Institute of Technology

Abstract

JavaCAP, a software tool for student authoring and searching of case libraries, has been implemented in support of the Learning-by-Design curriculum development project at Georgia Tech's EduTech Institute. Its case-authoring component, when used as part of the LBD curriculum, asks students to reflect on a recent Learning-by-Design experience, summarize it, and present important aspects of it and what they've learned from it for other students to learn from. By focusing on this part of the tool, we are able to both put our focus on supporting reflection and collect a library of student-authored cases that we will later edit and publish as exemplary cases for other students to use as models and to learn from.

JavaCAP is meant to be used as a collaboration tool. Supporting collaboration in middle school requires supporting both group work around the computer (synchronous collaboration) and asynchronous editing of cases. We support asynchronous collaboration by allowing each student to write on the case and providing distinguishing formats for each so it is easy to see the changes made since last using the tool and identify whose changes they are.

Our two studies suggest that JavaCAP does have potential as a collaborative reflection tool. We've used the metaphor of scenes in a play to help students remember different aspects of the experience they are analyzing and summarizing. Our first pilot showed that we had indeed found a way to engage students in effective reflection. However, this pilot also revealed flaws in the underlying technology. We needed to support asynchronous collaboration better than we were doing; the easiest way to do this was to reimplement using what the world wide web and its development tools had to offer. Our second pilot, using a revised tool both better supported asynchronous student authoring and made it easy for students to add images to their presentations. This study pointed out the need to better support writing itself within the framework we've created and suggests other collaboration features that students need.

Keywords— case-based reasoning, case-based learning, learning by design, LBD, scaffolding, middle school, science education, reflection, tools to support collaborative settings

A. BACKGROUND AND CONTEXT

A.1. Learning by Design

The context for JavaCAP's development is the Learning-by-Design (LBD) curriculum, currently being developed by Georgia Tech's EduTech Institute (Kolodner, 1996). LBD focuses on promoting better science learning among middle-grades students by having them learn science in the context of solving design problems. Students work in small groups to come up with ideas about how to address a problem, identify what they need to learn to solve it, and experiment, read, and explore to learn some of those things. They apply what they've learned as they solve the design problem. Building and testing their devices allows them to confront their conceptions head on. When a design fails, they recognize that there is some knowledge they are missing. Analysis of failure promotes identification of what they don't understand or misunderstand. The intent, then, though we cannot always make that happen, is that students will build and test their designs in order to gain immediate feedback about the physical laws they are trying to apply. Analyzing why something might not have worked as expected or might not have worked well enough and redesigning, rebuilding, and retesting are critical to LBD.

Students iteratively build, test, analyze, explain, and refine several times. Sometimes they build models of some aspect of what they are designing (e.g., when designing a model of artificial lung, they focused just on the pumping action (Hmelo, Holton, Allen, and Kolodner, 1996)). Sometimes they build a full model as a solution (e.g., when studying how to control erosion on an Atlantic Ocean barrier island, they built full models used sand, rock, moving water, etc.). During all activities, they focus on decision

making under uncertainty, generating several alternative solutions and comparing and contrasting them to each other.

LBD's pedagogical approach and many of its practices are borrowed from Problem-Based Learning (PBL) (Barrows, 1985). Important in both approaches is that students learn science concepts, cognitive and social skills, meta-cognitive thinking skills, and reflection skills as they are engaging in problem-solving and design activities. LBD, like PBL, emphasizes both doing and reflection on what is being learned. Reflection is an essential embedded activity in the curriculum, and students are asked to reflect on and articulate the full range of science concepts and complex skills they encounter as part of their problem-solving or design experience.

In addition, collaboration is perceived as essential for achieving meaningful design goals that are beyond the ability of the individual student. We are attempting to understand the best ways to orchestrate individual, small group, and whole-class activities to facilitate good reflection and learning in an LBD environment. And we are attempting to learn the best ways to facilitate communicative activities that allow that to happen, sometimes using technology (Hmelo et al., 1995) and sometimes proposing particular kinds of classroom activities that encourage sharing of ideas (Kolodner et al., 1995).

LBD is based, as well, on an analysis of the cognitive model underlying case-based reasoning (CBR) (Kolodner, 1993, 1996). CBR focuses on learning from one's own experiences and those of others. It tells us that one can use one's experiences well to the extent that one (1) has interpreted each experience well, (2) has access to one's old experiences, (3) can make the mapping between an old and new experience that allows something relevant from the old experience to be transferred, and (4) can apply and adapt what was learned from the old situation to the new one. CBR also suggests that one might use other people's experiences similarly if they are well-enough explained and available at the right times. We've all had the experience of gaining important insights through the examples and stories that others tell us.

This analysis suggests several things with respect to promoting children's learning:

- they will be more successful if we help them interpret their experiences well and understand what they can learn from them
- they will be more successful in reusing their experiences if we help them anticipate when something they have learned from an experience might be useful
- they will be more successful if we help them learn to recognize that they've had a previous relevant experience, help them map between old and new experiences, and help them apply and adapt an old experience for a new situation
- making the experiences of others available as children are solving problems might help them in all aspects of what they are doing and learning

A.2. Our tool: JavaCAP

It is within this context that our software tool, JavaCAP, is being developed. When complete, JavaCAP will include a case-authoring tool, case library, and library browsing and access tools appropriate for middle-grades learners (6th through 9th grades). Students will use JavaCAP to (i) publish what they have done and what they have learned and (ii) look up what others have published as needed while they are solving problems or designing. When populated, it will provide for students what a library of cases provides to experts: a rich 'database' of cases, stories, lessons learned, background information, and links between them (Kolodner, 1993; Domeshek & Kolodner, 1992).

JavaCAP is meant to be part of the larger system that comprises an LBD learning environment: sequences of problems that the students to solve; methodology they use to solve those problems; other classroom activities homework assignments; facilitation by the teacher; personal, paper, and electronic scaffolding; collaboration tools, reflection tools; and so on. It will serve as a reflection and articulation tool (when students are creating cases), provide advice to get students started solving problems (when they are using its cases), and help students learn to make connections between their varied experiences -- the kind of skill that is needed to promote transfer (Salomon & Perkins, 1987; Kolodner, 1996).

A.3. Current Focus: Reflection

Our focus to date is on student authoring of cases and what we need to provide so that students will be successful at (i) reflecting on their experiences and extracting from them what they have learned and (ii) presenting their experiences in ways that will allow others to understand them and learn from them.

While reflection is a critical part of learning, it is known to be very difficult for students (Rogoff, 1990). Our experience shows, as well, that students dislike reflection unless their teachers have introduced it in a way that allows them to see its use and that many teachers are uncomfortable with helping students reflect -- it is hard. Without reflection time embeddedinto the curriculum, students and teachers will tend to skip reflection by either continuing with

a design or problem solving effort itself or moving on to other activities.

Scaffolding is needed for both students and teachers to facilitate reflection. CSILE (Scardamalia, Bereiter, and Steinbach, 1984) provides the first major attempt at scaffolding reflection electronically. It scaffolds student discussions and arguments, asking students to articulate the purpose of each of the contributions they make to a discussion.

We needed to scaffold reflection in some different ways. First, CSILE's asynchronous collaboration and ways of integrating it into the curriculum seem unnatural for an LBD setting. Second, we saw that many teachers were uncomfortable with the way it took over the classroom. Third, we wanted something that would integrate better into the activities the students were doing (design) and support their reflection over a wide variety of complex skills, problem-specific issues, and science content. Fourth, we wanted to be able to integrate in some natural way the reflection that happens as one is solving a problem (almost always done but usually hidden) with the reflection that is so useful after solving a problem (to identify what has been learned, the kind of reflection that teachers and students are so uncomfortable with).

These two kinds of reflection seemed to us to require different kinds of tools. We've been designing design journals (Puntambekar et al, 1997) to help students make their reflections during problem solving and design activities explicit. But helping them reflect on the full experience seemed to us to require more than simply asking them to make reflections explicitly -- we needed to find a way, we thought, to make this kind of summative reflection feel like a natural activity and do it in such a way that students would quickly recognize the value of this kind of reflection.

This is where our familiarity with case-based reasoning and case libraries came to the rescue. We had seen earlier that students weren't much interested in case libraries that experts wrote for them (Hmelo et al., 1997). But we knew also, from our work with undergraduates, that authoring of cases worked well in promoting learning and that students liked reading each others' cases. We decided to try this with middle schoolers. This is how the concept of JavaCAP was born.

B. JavaCAP

JavaCAP evolved from case libraries developed at Georgia Tech for aiding experts (Domeshek & Kolodner, 1992), aiding architecture students, and for supporting engineering and industrial design education. A case library for experts holds "cases" dedicated to a specific content area. Before getting involved in educational endeavors, our AI group at Georgia Tech worked with architects and engineers to develop case libraries of prisons, courthouses, skyscrapers, public libraries, and handicapped access to support architects; for hydraulics engineers working on aircraft hydraulic system design; and so on. An expert using a case library might browse it to get ideas or ask to see cases that address a particular issue (e.g., lighting in a judge's chambers, effect of high pressure on a hydraulic system). Cases can suggest ways to address a problem, suggest solutions, suggest results that might accrue, and so on.

We've also successfully made these case libraries available to students in our architecture and industrial design classes. But, as stated above, we had problems when we introduced expert-written cases to middle school students (Hmelo et al., 1997). The cases, as we wrote them, were not aimed correctly for them. But more than that (we could have written them better), they simply weren't interested in reading what experts had to say. By asking students to author cases for others to use and then asking students to use other students' authored cases, we thought we could add both an element of engagement that was missing in our adult-written case libraries, an authentic reason for students to summarize and extract what they learned, and an activity for students that would promote deeper learning. We've focused first on case authoring.

A big issue we had to deal with was how to help the students look back on what might have been a two-week or longer project and extract what they had learned. We've dealt with that issue two ways. Students keep design diaries (Puntambekar, et al., 1997) throughout their activities. Design diaries differentiate between different phases of their design activities: understanding the problem, generating and choosing between several possible alternatives, building, trying out, and analyzing the solution, and so on. Our first way of dealing with this issue is that we've organized JavaCAP in the same way and asked students to have their design diaries in front of them as they are writing their summaries. When entering their experiences understanding the problem, they know which design diary pages to refer back to. Our second solution is as yet unimplemented: we intend to integrate electronic versions of design diaries (which support reflection while working on a problem or project) with JavaCAP (used after completion of the project).

JavaCAP asks students to think of their previous design activity as a play with scenes. They compose a case by describing their experiences as if they are writing a narrative (not quite a play, as they are describing and summarizing what happened rather than writing dialog). Students can upload multimedia elements for a scene and annotate them. When finished, they can publish their case by choosing indexes for it, anticipating the kind of

description each of their scenes needs to have for others to recognize its usefulness to their endeavors.

JavaCAP is implemented using Java and runs over the World Wide Web (hence its name). JavaCAP's main and authoring menus are displayed in Figure 1. The main menu at the left bar allows students to browse, search or author cases. Also, customizing is available for authorized users. When authoring is chosen, the student logs in with his/her user name and tells the system which design problem s/he is commenting on. From the authoring menu students may access the four scenes or publish their case.

B.1. The Scenes

The first scene in JavaCAP is the "Problem Presentation" scene, where the problem specifications and expectations are presented (see an example in Figure 2). Then comes the "Alternatives Selection" scene where students complete an alternatives table, entering the criteria used to compare the alternatives considered (see an example in Figure 3). In the "Solution" scene they describe the solution they chose, what happened as a result of trying it out, and what they learned from it. Finally, they conclude the entire experience in the "It's a Wrap" scene, where they can also describe anything else that was especially important to their effort (e.g., collaboration experiences, what they learned about doing design).

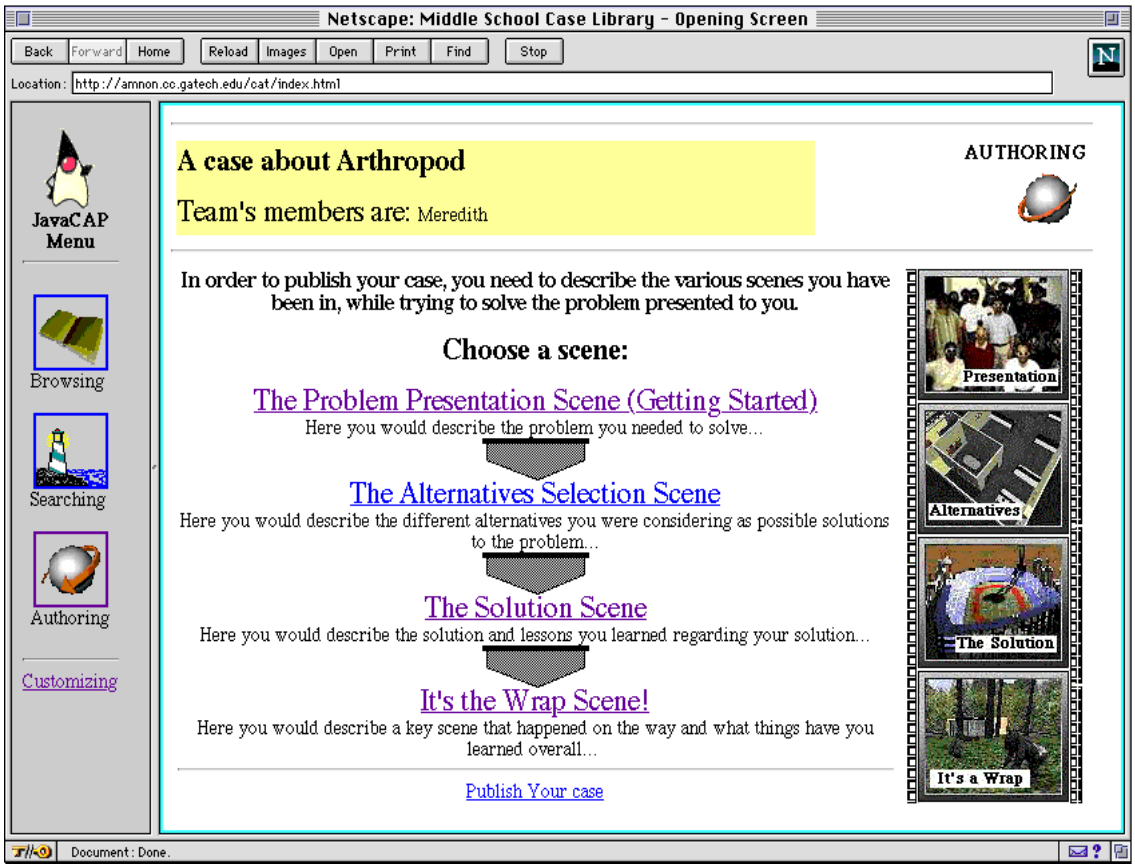

Figure 1: The authoring menu of JavaCAP after logging in through "Authoring" at the left bar main menu.

With the exception of the alternatives scene, all scenes are designed with the same 'two-column notebook' screen. The left column of the notebook provides an unstructured area for kids to freely write about their experience, while referring to an image. Students use digital cameras to capture their design artifacts. Annotating images anchors the reflection process to concrete and attractive elements. On the right column of the notebook there are labeled slots for text entry as well as pull-down menus for indexing. This format creates a structured and sequenced procedure for the reflection process. Some

of the slots are accompanied by scaffolding features such as suggestions for criteria (general and problem-specific) to compare alternatives or lessons learned in previous cases (Guzdial, 1994).

The two-page notebook interface supports collaboration by providing distinctly different space for two approaches to expressing the problem and the collaborative design solution. For instance, in the "Problem Presentation" scene, the right-hand page includes requests for a title for the case, the problem presentation and some learning expectations. On the left-hand page, the students are simply prompted to talk about what was important to them when they began the design problem. They are encouraged to use a digital image to illustrate some aspect of the this initial phase of the design process. This duality requires distinctly different responses, providing collaboration based on an image and again on more structured text responses. While one student may contribute by drawing a scale diagram of the proposed solution, another may be best able to synthesize the group's statement of the problem. The paired presentations elicit and support multiple levels of student participation in the group solution.

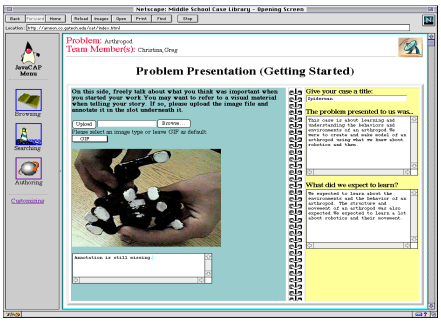

Figure 2: The 'Problem Presentation' scene in JavaCAP.

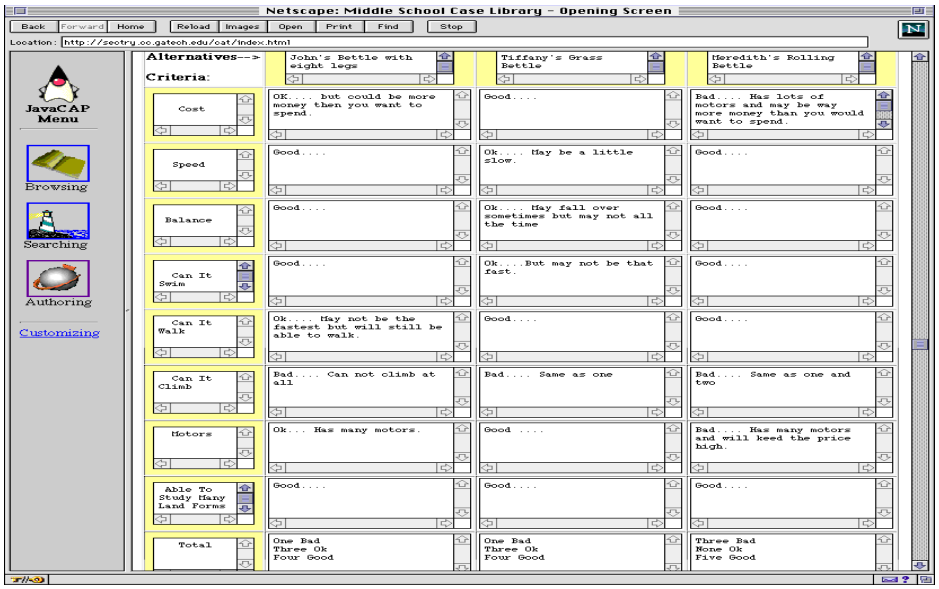

Figure 3: The 'Alternatives Selection' scene in JavaCAP where a student filling a 9x3 table.

Figure 3 shows an "Alternatives Selection" scene. This is the most challenging scene. Kids summarize the set of criteria they used (the vertical dimension of the table) to make their design decisions and what alternatives they considered (the horizontal dimension). During design, they might or might not have pulled together all of their considerations in this way. We ask them, in their summaries, to do that. Many groups have "design fixation", finding it difficult to go beyond their initial idea. To encourage consideration of multiple design alternatives, the students are asked to evaluate at least two solutions against the same set of criteria. The interface in the "Alternatives" scene illustrates this alignment of different solutions, side-by-side in vertical columns against the same set of criteria. Students have difficulties understanding the concept of criteria used to select the best alternative. We ask them to give it another try after solving a problem even if they had trouble doing it while they were engaged in design. JavaCAP provides prompts for criteria appropriate to each problem and also describes criteria in a generic manner. Our intention is to integrate the scaffolding for summarizing (in JavaCAP) with scaffolding for making the design decision using electronic design diaries.

In the two summative scenes, i.e., the "Solution" scene and the "It's a Wrap" scene, there are several pull-down menus that aid students in indexing their case based on the knowledge accrued in the case library. For example, students need to select from a list of problems found in previous solutions, the one that is most similar to the main problem they have encountered in their solution. The list is compiled and customized by experts and teachers, based on the growing database of cases (see a further section on customizing).

The "It's a Wrap" scene is where the group can describe important aspects of the design solution or process. The free-form page of the notebook provides an area for the group to summarize their experience or tell their story and illustrate it with an image. The other page guides the group through describing a key scene from their work. This significant event may be the location of an important resource or some aspect of the collaborative experience. There is a pull down menu to select an activity or tool that was used in this "key" scene for the design process. The group goes on to describe what happened in that crucial event and then what they have learned in general from working on this problem. This scene provides the "creative" outlet for the group. "It's a Wrap" is the place where they get to "rap". It is like a prologue, where the group has an opportunity to reflect on what is significant at the end of the design process. They can contrast this to their initial expectations in the "Presentation" scene. When peers use the case, these personal reflections may provide valuable insights to the process as well as the content of the design solution. The last scene is very open-ended,providing a forum for reflection on the entire spectrum of the processes and content experienced in the design problem.

B.2 Supporting Collaboration

We envision the use of JavaCAP as a collaborative tool, where authoring a case is done by a team of students. One team approach is to assign each member to be responsible for the first draft of one scene, all members review each scene, then collaborate on the final version of every scene. JavaCAP facilitates this collaborative work of a team in an asynchronous mode. Once a team member has entered a scene page to edit, no one else in his/her team can enter that page. The warning message includes a link to the e-mail of the member who is currently editing the page, as well as other contact information. When a team member exits the scene page, then it is unlocked for others to edit. Team members can review their peers work at any time, entering any of its scenes through the browsing section. Of course, there are many collaboration issues that need to be addressed before we can say that JavaCAP provides mature collaboration support.

B.3 Supporting Customization

In order to be able to include prompting appropriate to particular design problems students are reporting on, JavaCAP has a facility for customization that is simple enough for teachers to use. Teachers can refine existing prompts and scaffolding features, or even add new materials that might better fit their students' needs. The general preferences include modifications to problems and its types, activities/methods/tools, solution statuses, and suggestions for criteria by problem types. For a specific problem the preferences include: suggestions for criteria, classifications of the solution, and major problems encountered in the solution.

C. EXPERIMENTS

The authoring module of JavaCAP was tested twice in a Georgia middle school to evaluate the usefulness of the authoring tool and to acquire student authored cases.

C.1 Piloting with Students

An early version of JavaCAP was used by seven eighth grade earth science students and one seventh grade life sciences student, selected by the classroom teachers from a pool of over forty-five volunteers. The students used their individual design diaries to assist in authoring cases representing the experiences of their group and its chosen solution.

The volunteers joined the EduTech staff members in a science classroom after school for two sessions (a

total of three hours) in November, 1996. The authoring tool was accessed via an intranet of nine PowerBooks, with one as the server.

The students were able to complete their cases and rated the tool's usability as very good. They liked the permanence of the case in the computer database and publishing their solution.

C.2 Class Use with Students

A new and improved version of JavaCAP was used later by thirteen seventh grade life sciences students in June, 1997. They had just spent three weeks in class developing a group design for robotics locomotion derived from the way one or more arthropods get around. The students worked in groups of three to five keeping individual diaries of their design work as well as a group portfolio of their experiences and design products. These students felt comfortable with the JavaCAP interface since they had used another EduTech software tool with a web-browser interface. The teacher selected thirteen students from two classes, representing a cross-section of abilities and interest levels. In each class there was one pair of students who had worked within the same team, each of these pairs authored a single solution collaboratively, the other students authored individually.

The selected students joined EduTech staff members in the middle school media center during their science class on two successive days. JavaCAP was accessed via Internet connections available in the school; it was served from a Macintosh server at Georgia Tech. None of the students had previous experience with JavaCAP.

Students were provided with their individual diaries from the problem and their design artifacts from the portfolio. They took turns capturing images of their design products with a digital camera. Images were captured during the first session. Overnight, EduTech personnel loaded them onto the web into the scenes students had designated so that students could write in the context of the images on day two.

Students were excited to author their recently-completed problem solutions and were able to use the software with minimal instruction. This was the first test where the uploading of images was a working option. Students found the digital camera easy to use and were able to efficiently acquire images to accompany the case they were authoring. The four-scene framework for recording their solutions was readily understood. Students grasped the different spaces within the authoring tool and were able to move between the scenes as intended. Most completed the scenes in the order presented on the introductory screen, but a few skipped a scene in their authoring. Of the eleven cases authored, only three had alternative tables providing information for several solutions and a systematic evaluation mechanism. The "Solution Scenes" contained general functional science knowledge, many described some detail of the collaborative process they used to solve the problem. The final scene, "It's a Wrap," was most often used to show the importance of research to their solution and the collaborative process in determining the solution. These initial student cases contained only high level functional descriptions of science (we would have liked more technical detail), but the play metaphor elicited some very specific explanations of the collaborative design process students used.

D. DISCUSSION

D.1 Lessons Learned

The net-browser interface for text and images is an easy interface for students to use. The middle school students readily fill in the blanks on the display form, but the content of the cases contained very little evidence of science learning in the problem domain. The pull down menus within the text areas were almost always used, but hyperlinks to scaffolding were not. Most of these links were at the bottom of the display, and may not have been seen in time to be used. The time allocated to student authoring will need to accommodate the accessibility of the cases on the network and the complexity of collaboration. Accessing the tool via the world-wide web yielded variable response times. The class test did not allow enough time for all students to complete their cases or to review and edit them. Image capturing and uploading adds a considerable amount of time to the case authoring. However, the use of images, diagrams and models to represent a design idea is very valuable in the authoring process, the additional time is well-spent.

Even though nine of the eleven test cases were authored individually, the students often portrayed the collaborative nature of their solutions. Several scene titles included the group name and many alternative solutions were identified by student names. The more specific entries in the scenes usually portrayed collaborative processes, with either favorable results or stating how it might work differently in the next design problem. For instance, an entry in the "It's a Wrap scene" by one seventh grader:

> "Working in a group and putting your heads together gets things done faster. We have also learned a lot about the movement of the arthropods that we studied."

During the individual authoring, the students naturally gravitated to the computers in pairs working together on how to use the case tool. The synchronous collaboration is very evident in the students' case authoring, while the asynchronous mode is not so familiar to them.

D.2 Issues to Explore

Initial development and trial of JavaCAP perhaps raises more questions that it answers. Some have to do with its use. Will JavaCAP perform well when the team members access the case asynchronously to author and revise the scenes? How will teachers use the tool in the curriculum and how will they manage the computer access required to author cases? What is a reasonable time frame for the middle school student to be able to author? Will the depth of science knowledge in the scenes increase if more time is allocated to authoring? Will more iterations of authoring increase reflection and thus lead to better construction of scientific knowledge?

Others have to do with the potential contributions of case libraries and case-authoring tools to learning. When will browsing the middle school student authored case library be useful and in what ways will it strengthen the student's ability to grapple with an ill-structured problem? Should there be criteria for including student authored cases? Can the student learn about the design process by browsing a case authored by a design team, even if the domain knowledge is distinct from that of the current problem? Can the case reflect the process as well as the content used to derive a solution?

Some have to do with its integration into the classroom and LBD environment. When is it better to start using JavaCAP along the design process? Would it be appropriate to author the problem presentation and alternatives selection scenes when the design solution was chosen, then finish the solution and it's a wrap scenes after the design is completed?

But the most important questions, we think, have to do with the role the software should play in learning and the embellishments it needs to help students summarize and articulate better. What they entered into JavaCAP, as mentioned above, was lacking the technical kinds of details and specificity we would have liked. The problem may not be with JavaCAP; it may be with what students were pushed to do during their classroom experiences. We've found that, in general, students are not pushed to get to details in middle school but that when we sit with them and talk and prompt, they are able to consider far more than is asked of them. We will have to find out what levels of detail we can expect and how to promote it.

More broadly, we would like to be creating tools that, in some sense, free the teacher from responsibility for facilitating perfectly. If a teacher is weak at helping students identify constraints and criteria or doesn't know what kinds of questions to ask students to get them down to specifics, for example, a piece of software that scaffolds appropriately might fill in. Other times, teachers and software will have equal expertise, with the software and teacher supporting each other. Other times, the teacher will be able to help more. We see a symbiotic relationship here. When the software is more capable than the teacher, it might serve as scaffolding both for the students and the teacher. The teacher may learn from the software how to get the students to more specificity, for example. When the program is weak compared to the teacher, we will know that we need to examine what teachers are doing and decide whether this is something that we want or can feasibly put into our software.

A tool that would help students be more specific scientifically might also help teachers create more interdisciplinary curriculum. Students wereconcerned with spelling and punctuation, the need to edit their draft case, and the desire to be understood by many. A tool that helped students with the science would make it easier for a language arts teacher to help students write a narrative describing a science experience.

E. SUMMARY

We have described JavaCAP, a case authoring program we developed aimed at middle-school students studying the EduTech LBD curriculum. Middle-schoolers can author cases in teams and use the tool in a collaborative way to share their experiences while learning science through solving design problems. Their teachers can customize scaffolding and indexing features in JavaCAP to better fit their students' needs. Students can upload multimedia elements they have captured during their studies and eventually they can publish their cases on the Web.

We have tried JavaCAP with 20 students and made significant changes in its structure and interface along the formative evaluation process. According to our observations, the program is usable and the current interface is well understood by students and teachers, though some concepts remain difficult for students, such as distinguishing between criteria and alternatives. Teachers collaborating in the development of the LBD science curriculum are authoring exemplary cases and customizing JavaCAP for new problems they develop for their students. Further research will be completed in order to learn how to better scaffold the authoring process and to enhance the collaboration features.

Future research efforts will focus on use of JavaCAP to support teaching in collaborative settings at the middle school level. The tool provides the opportunity for teachers to form student groups based on interests, rather than the by classroom period. JavaCAP would provide the collaboration across class periods or even between schools, where the curriculum implementation called for wider-based collaboration. JavaCAP might provide the forum for

the sharing of ideas, another tool from our research group may be employed to facilitate the cross-classroom discussion.

Another focus will be to determine how JavaCAP can better facilitate reflection and iteration. The authoring portion of JavaCAP is designed to be completed at the conclusion of the design process, to prompt the students to take another look at what they have learned and accomplished within their group. But we see JavaCAP playing an important reflective role throughout the students' design and problem-solving processes. The task of writing a case would be easier on the students if it was a piecemeal, iterative process, rather than a colossal task at the end of a project. Further, the solution and "It's a Wrap" scenes could be more reflective. It is important for students to remember and record their design process, including wrong-turns and other realistic elements that might not appear in a statement of a final, clean solution. We need to find a way to encourage that.

One path to a greater role for the JavaCAP software is to continue our process of integration between JavaCAP and other collaborative tools being developed by the EduTech Institute (Mark Guzdial et al, 1997). For example, JavaCAP has many elements that would dovetail nicely with the design diaries that we have been developing, and use of JavaCAP could be easily scaffolded through use of the Planning tool in SMILE.

In summary, JavaCAP is playing an important role in encouraging student reflection through collaborative case authoring. The task of case authoring is challenging for students, as are many learning activities. Our hope is to improve integration and guidance (through scaffolding) to ease the complexity and improve the learning benefits of the task.

Acknowledgments

The research reported here was supported by the Woodruff Foundation, the ARPA CAETI Program under contract N66001-95-C-8608, and the McDonnell Foundation.

References

Barrows, H.S. (1985). *How to design a problem-based curriculum for the preclinical years*. NY: Springer.

Domeshek, E. A., & Kolodner, J. L. (1992). A case based design aid for architecture. In J. Gero (Ed.), Proceedings of the Second International Conference on Artificial Intelligence and Design.

Guzdial, Mark, Cindy Hmelo, Roland Hübscher, Kris Nagel, Wendy Newstetter, Sadhana Puntambekar, Amnon Shabo, Jennifer Turns, and Janet L. Kolodner (1997). Integrating and Guiding Collaboration: Lessons Learned in Computer-Supported Collaborative Learning Research at Georgia Tech (CSCL '97), Proceedings for Computer Support for Collaborative Learning 1997.

Guzdial, M. (1994). Software-Realized Scaffolding to Facilitate Programming for Science Learning. *Interactive Learning Environments*, 4(1), 1-44.

Hmelo, D. L. Holton, J. K. Allen, and J. L. Kolodner. (1996). Designing for Understanding: Children's Models of Lungs, In *Proceedings of* the *Nineteenth Annual Meeting of the Cognitive Science Society*. Mahwah, NJ: LEA.

Hmelo, C. E., Vanegas, J. A., Realff, M., Bras, B., Mulholland, J., Shikano, T. and Guzdial, M. (1995). Technology Support for Collaboration in a Problem-Based Curriculum for Sustainable Technology, in *Computer Support for Collaborative Learning* (CSCL '95), J. L. Schnase and E. L. Cunnius, Eds. Bloomington, IN: Lawrence Erlbaum Associates, pp. 169-172.

Kolodner, J., and the EduTech Design Project. (1995). Design Education Across the Disciplines. Paper presented at the Second Congress on Computing in Civil Engineering, Atlanta, GA.

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

Kolodner, Janet L., (1996). Educational implications of analogy: A view from case-based reasoning. *American Psychologist*.

Puntambekar, S., Nagel, K., Hubscher, R., Guzdial, M., and Kolodner, J. L. (1997). Intra-group and intergroup: An exploration of learning with complementary collaboration tools. In *Computer Support for Collaborative Learninb (CSCL '97)*. Lawrence Erlbaum Associates. (this volume).

Rogoff, B. (1990). Apprenticeship in Thinking: Cognitive Development in Social Context. New York: Oxford University Press.

Salomon, G., & Perkins, D.N. (1989). Rocky roads to transfer: Rethinking mechanisms of a neglected phenomenon. *Educational Psychologists, 24*, 113-142.

Scardamalia, M., Bereiter, C. and Lamon, M. (1994). The CSILE Project: Trying to bring the classroom into World 3," in *Classroom Lessons: Integrating Cognitive Theory and Classroom Practice*, K. McGilly, Ed. Cambridge, Mass.: MIT Press, pp. 201-228.

Author's Addresses

Amnon Shabo, Kris Nagel, Mark Guzdial, Janet Kolodner: EduTech Institute, Georgia Institute of Technology, GCATT Building, 250 14th Street, N.W. Suite #138, Atlanta, GA 30318-0490. amnonh@cc.gatech.edu, kris@cc.gatech.edu, guzdial@cc.gatech.edu, jlk@cc.gatech.edu.