Authoring Tools for Learning Suites to Support Contextualized Computing Education

PI: Mark Guzdial
May 2005

Project Summary

Experience in prior NSF supported work has shown that (a) embedding computing instruction in a context that students find relevant to their future professional lives leads to improved retention and motivation in computing and (b) learning suites can serve to scaffold student learning in science inquiry. We are proposing the creation of tools for inexpensively creating learning suites for students in computing with a particular focus on offering more contextualized learning opportunities to students. Our goal is to create tools that are no harder to use than word-processors and PowerPoint, but generate learning suites that include collaboration, scaffolding, and prompts for reflection and process guidance.

Intellectual Merit: The proposed work contributes to our understanding of the role of technology in creating learning experiences tailored to students’ interests, specifically in the area of Computing. The technology research questions being studied are on how we can support low-cost authoring of learning materials that integrate across media and time. The learning science research questions are on how inexpensively authored learning materials can contribute to motivation and learning by trading off contextualization for high production values.

Broader Impacts: The potential impact is to draw more students into STEM disciplines by offering them contextualized, relevant, and motivating learning opportunities. While the current work is situated in computing, the questions being explored and solutions being developed are as relevant to mathematics, science, and engineering fields. Since the learning techniques being developed have been used successfully to draw in women into computing, the proposed work may help in drawing more diversity into other STEM disciplines.
Authoring Tools for Learning Suites to Support Contextualized Computing Education

Contents

1 Introduction: Contextualizing and Tailoring Computing Education ........................................ 1
   1.1 The Value of Context to Tailor Computing Education ..................................................... 1
   1.2 Learning Suites for Undergraduate Computer Science Education ................................. 3
       1.2.1 Relevant past NSF experience ................................................................................. 6

2 Project Plan .............................................................................................................................. 7
   2.1 Prototype Development Plan ............................................................................................ 7
   2.2 Evaluation Plan: Settings and Questions .......................................................................... 10
   2.3 Dissemination Plans ......................................................................................................... 11

3 Conclusion and Project Schedule ........................................................................................... 11

4 Budget Justification ............................................................................................................... 13

5 Facilities .................................................................................................................................. 14
Authoring Tools for Learning Suites to Support Contextualized Computing Education

1 Introduction: Contextualizing and Tailoring Computing Education

Across SMET disciplines, but especially in CISE-related disciplines, we have a significant problem in motivating and retaining students [Foundation, 2000] [Foundation, 2002]. We have found that contextualizing computing education has a dramatic impact on retention and motivation [Rich et al., 2004] [Forte and Guzdial, 2005] [Tew et al., 2005]. By contextualizing, we mean presenting the course content (e.g., examples, assignments, even lectures related more to the context than computing) in terms of a pervasive problem domain that student find relevant.

At Georgia Tech, we offer three different contextualized introductory computing courses. That’s quite expensive for us, and we’re one of the largest computing academic units in the United States—and yet, there’s interest among some academic units in the College for us to offer even finer-grained contextualization of computing courses. How can we avoid offering multiple CS1’s and still offer contextualization that students find relevant? How can even smaller departments offer contextualized instruction in computing, where the issues of retention and motivation are just as pervasive?

We believe that an answer may lie in creating and providing to the students on-line materials that support learning in different contexts, perhaps used as an alternative path in the middle of a traditional face-to-face course. But we don’t want to just put PowerPoint slides on-line. We know from the literature on distance education that student retention while learning with on-line materials is the biggest challenge for distance education [Bernard et al., 2004], and that higher production values in the material is highly correlated with better retention [Daniel, 1996]. We know from our own experience that a successful contextualized course involves a host of learning resources and student learning activities [Forte and Guzdial, 2004]. To be successful, we have to provide a rich set of learning experiences, but do it without high cost. We propose the development of tools for creating and packaging (for sharing) of on-line learning suites that include learning materials informed by learning assignments, student learning activities (e.g., homework assignments), opportunities for collaboration, relevant learning cases, and other resources. We propose to evaluate these tools to explore (a) their costs and usefulness to authors (faculty) and (b) the impact on retention and learning of students.

1.1 The Value of Context to Tailor Computing Education

“We must broaden our horizons and think of our students not only as potential compiler or operating systems designers but also as implementors of computer-based solutions to non-computing problems.” Sharon Lawrence Pfleeger in [Pfleeger et al., 2001].

In our work on creating introductory computing courses that motivate women and members of other under-represented groups\(^1\), we have found that contextualizing the computing education has had a dramatic impact on student retention and motivation [Guzdial, 2003] [Forte and Guzdial, 2004] [Rich et al., 2004] [Tew et al., 2005]. By contextualizing, we mean that the course focuses on the learning computing skills and concepts in terms of concrete applications for solving real domain problems. Iteration, data structures, and conditionals are not presented as abstract topics in the first two courses, but in terms of their use to process media for communication [Guzdial and Soloway, 2003], or in terms of processing data in engineering applications [Smith, 2005].

Georgia Institute of Technology (Georgia Tech) is predominantly an Engineering school with highly rated programs. Overall, Georgia Tech is about 24% female. In Fall 1999, the Faculty required all students at Georgia Tech to take a course in computing. Until Spring 2003, only one course met that requirement, and all students took it. The average success rate was 72%. Georgia Tech now offers three different CS1 classes:

\(^1\)Funded in part by NSF CISE Educational Innovations grant #0306050 and Division of Undergraduate Education grant #0231176
• One contextualized in media that is mostly taken by Liberal Arts, Management, and Architecture majors [Guzdial, 2004] (typically \( n = 300 \) each semester at Georgia Tech). This course introduces programming and computing concepts through manipulation of images (e.g., cropping, scaling, negating, reducing red-eye), sounds (e.g., splicing and reversing), HTML (e.g., parsing headlines out of a news page, generating index pages), and video (e.g., implementing chromakey effects).

• One for Engineering majors that uses MATLAB, the lingua franca of most engineering disciplines [Smith, 2005] (typically \( n = 1000 \) each semester) and draws its applications from engineering domains.

• One using a traditional CS1 text and approach (e.g., focusing on numbers and strings as data types without a problem context for the whole course) (typically \( n = 125 \) each semester). This course is mostly taken computer science and College of Science (e.g., mathematics, physics, psychology) majors.

Contextualized introductory computing courses at Georgia Tech have had dramatically better retention than more traditional CS1 courses [Rich et al., 2004] [Forte and Guzdial, 2005], e.g., an increase from a 72% success rate (students completing the course earning an A, B, or C) to a 90% success rate in the media computation course, with similar results in the engineering course. Similar results have been found at a two-year college adopting our media course [Tew et al., 2005].

In particular, contextualized computing courses are successful at retaining women than traditional introductory computing (CS1) classes. In our media computation class, women succeed at the same rate as men [Tew et al., 2005] and report finding the class to be relevant, motivating, and worth continuing study [Rich et al., 2004]. In the Spring 2005, our follow-on computing course was 75% female.

The impact of the contextualized courses and greater diversity extends into more explicitly computing-related degree options. The CS major at Georgia Tech is typically 9–12% female. In the last year, we have begun offering a CS minor, so that students will be encouraged to take more computing within their major, and a new BS in Computational Media, joint between the College of Computing and the Ivan Allen College of Liberal Arts at Georgia Tech. The CS minor now has two dozen students declaring an intention to take the minor, mostly from Engineering and Liberal Arts and 16% are female. The Computational Media major, without any external advertisement (as it was just approved), already has 58 majors (as transfers from other majors), a quarter of whom are women.

The importance of context in computing education shouldn’t surprise us. We have known for decades that the knowledge of professional software engineers is domain-specific—they can write software with expertise within a domain that they have experience, but in a different domain, they have much greater difficulty writing software of similar complexity [Adelson and Soloway, 1985]. We also know that studies of why women have not participated in computing have emphasized that female students often find computing courses “irrelevant” [AAUW, 2000], emphasizing details that are not related to solving any particular problems [Margolis and Fisher, 2002], and offering little opportunity for solving socially relevant problems in creative ways [Pfleeger et al., 2001]. A context can be used to frame computing in terms of solving relevant domain problems, which students do report finding creative and motivating [Rich et al., 2004].

Does it matter that students aren’t learning more abstracted computer science anymore? Yes, but all the indicators suggest that it’s better not worse. As Pfeegler points out at the beginning of this section, the next generation of computing jobs will be in applying computing in non-computing domains, not in traditional computing areas. Just being inward looking won’t help future information technology professionals. The U.S. Department of Commerce reports that there is actually an increasing demand for information technology professionals in the future, but these may not all be traditional programming jobs\(^2\). Further, there is evidence from the learning sciences that learning

well within a context may lead to learning that may better transfer to new contexts later [Bruer, 1993] [Kolodner, 1997] [Linn and Hsi, 2000].

Contextualized computing education seems to be of benefit to students, but how much contextualization is useful? Should we be creating a different CS1 for each major or for each common set of students’ interests? One would hope that that shouldn’t be necessary, simply because of the expense involved. Are three different CS1’s enough to cover the modern academy? Based on conversations with our customers, both students and faculty, they are much happier with three choices than one, but there are requests for greater contextualization. For example, the College of Management would like a CS1 that uses Excel as the programming language and whose domain applications would include database manipulation. The School of Electrical and Computer Engineering would like us to use C in the first course with more ECE-like applications. While we can undoubtedly teach CS1 learning objectives within that context, teaching yet another course is expensive for the College of Computing–and we’re one of the largest CS academic units in the country with some 70 faculty and over 1300 majors. How can small computing departments possibly offer contextualization to meet demand?

What is working for computing could conceivably work for other disciplines. For example, there have been calculus and algebra versions of physics offered at many institutions, but that’s not contextualization in the way that we’re describing it here. Physics could be taught using applications from civil engineering and biomolecular engineering, and while the physics would be the same, the difference in applications could be significant in terms of student retention and motivation. But the economic problem remains of offering so many different versions of courses in physics, mathematics, and other foundational subjects.

1.2 Learning Suites for Undergraduate Computer Science Education

We recognize that our contextualized computing education is successful not just because of the contextualization. Each of these courses is actually a rich collection of learning resources and activities that are assembled by teachers. Our students tell us that the collaborative website used in the course, CoWeb or Swiki [Guzdial et al., 2001a], plays a big role in their enjoyment of the course [Rich et al., 2004]. In one of our unpublished surveys, 20% of the media-CS1 students surveyed identified the course space as the “best” part of the course.

Let’s consider some of the learning resources are made available to students in the media computation CS1.

- **Learning Resources**: As in most classes today, the media computation class has both a textbook [Guzdial, 2004] and on-line access to all lecture slides (Figure 1).

- **Process Guidance**: Students in a computing course are engaged in project-based learning, and learning sciences have identified some of the significant supports that are necessary to make students successful at project-based learning [Blumenfeld et al., 1991]. One of these is guiding the process–helping students to figure out what to do next and where to go for help. The media computation CoWeb uses some of the techniques that learning scientists have identified for guiding process [Quintana et al., 2002], such as providing a list of “Hotspots” that are regularly updated identifying the current pages that are most useful right now, and where to go to ask questions about what’s going on right now (Figure 1).

- **Anchored Collaboration**: Homework assignments are available in the CoWeb, as are exam review pages with sample problems. But all of these resources are structured as anchored collaboration [Guzdial and Turns, 2000], where one click away are the collaboration spaces to post answers, see others’ answers, ask questions about the review question or the homework, and have a focused discussion about the homework or review (Figure 2). We have found that anchored collaboration results in longer on-topic discussions than traditional threaded discussion boards [Guzdial and Turns, 2000].
Figure 1: The front page of two course CoWebs, showing some of the guidance provided, including day-to-day guidance.

- **Case libraries**: Students are encouraged to post the media that they create in the CoWeb **galleries** for sharing with others (Figure 3). The act of sharing the results of one's program in an introductory course turns programming from a solitary activity to a social one whose goal is the creation of interesting media to share [Forte and Guzdial, 2004] [Rich et al., 2004]. In other CS courses, the CoWeb is used to share complete programs in a case library where the goal is not to share work with the current cohort, but with the next cohort [Guzdial, 2001b] [Guzdial et al., 2001b]. Students find the past work in the galleries or cases pages very useful in terms of setting expectations, understanding the scope of the problem, and considering alternative paths to solutions [Rich et al., 2004] [Guzdial and Kehoe, 1998].

We consider this collection of learning resources and activities to a learning suite, like those that have been created for science inquiry activity by middle school students [Quintana et al., 1999] [Quintana et al., 2002]. These are different from the work in learning objects [Singh and Bernard, 2004] [Laleuf and Spalter, 2001] in that we are not automatically assembling these, nor defining metadata to describe them, nor providing search mechanisms for identifying new learning resources and activities. Rather, we are implementing a set of learning resources and activities in an orchestrated...
Our proposed work is to make it easier to create these kinds of learning suites for contextualized undergraduate computing education. Our hope is to make it easy enough that some faculty will be willing to assemble collections like these and then package them (with our tools) so that they can be adopted and tailored by other faculty.

We are seeing that kind of sharing now. Robert Sloan of the University of Illinois-Chicago has an NSF DUE Adaptation and Implementation grant to create two versions of the media computation CS1, and he has now adopted the CoWeb (e.g., galleries, Q&A pages, etc.), too, as a critical piece for the success of the course. We are also seeing the importance of this infrastructure. Some departments who have adopted the media computation approach but who have not implemented the learning suites (e.g., no on-line galleries) have not had the same retention benefits.

With the kinds of supports that we’re calling learning suites, it becomes easier to offer a contextualized computing course. That’s copying the model that we’re currently using at Georgia Tech. But we’re also planning to explore going one step further. Is the undergraduate learning suite powerful enough in its support of learning that students can succeed with it, and fewer lectures? If so, could the same instructor actually be teaching several contextualized courses at once, each supported by an on-line learning suite? Thus, we can offer even more contextualized courses (or segments of course) with existing resources.
It may be possible to teach non-contextualized computing courses with our learning suites, but that’s not the research question of interest to us. Our goal is not to create alternatives to brick-and-mortar schools, but to create better learning effectively and economically. We are exploring learning suite components that are work well with contextualized computing courses, and they may not work as well in non-contextualized courses.

Our plan is to develop authoring tools for learning suites, and then study:

- **Author-Teachers:** Is it easy enough? Are we providing the right learning resources and activities? How difficult is it to teach with a learning suite? How much additional effort is it to teach multiple contextualized courses, each supported by a learning suite?

- **Students:** We know that on-line resources most often lead to higher rates of attrition and don’t always have the same learning results [Bernard et al., 2004] [Daniel, 1996]. Are these true for learning suites, too? Further if students are being supported more by a learning suite and less by lecture (i.e., as we move closer to a pure on-line model), as we are proposing to explore, does retention and learning decrease?

### 1.2.1 Relevant past NSF experience

Guzdial was a co-PI on the *Computer Modelling for Curriculum Integration* project, funded by the NSF REPP program (REC-9814770). That project sought to use computer-supported collaborative learning to integrate the various curricular elements (computer science, mathematics, and engineering) that lead to students’ understanding of computer modelling. While we had success in some domains using our collaboration support (e.g., in some computer science classes [Guzdial, 2001b]), it was generally not a successful activity. Our finding was that the culture and classroom practices of mathematics and engineering courses inhibit collaboration [Guzdial et al., 2001a]. Our focus then shifted to the faculty, working with them to develop collaboration tools that meet their goals [Guzdial et al., 2000] which resulted in a tool (the CoWeb for Collaborative Website, and also known as *Swiki* for Squeak-based Wiki [Cunningham and Leuf, 2001]) that has surprisingly good adoption (and even invention of new activities) by faculty into their classes [Guzdial et al., 2001b]. As of this writing, a Google search of “swiki” results gives 219,000 hits with use literally all over the world. We have shown the advantages of a computer-supported collaboration approach.

---

5[^1]: [http://coweb.cc.gatech.edu/swiki](http://coweb.cc.gatech.edu/swiki)
in a comparative study in English composition in which CoWeb-using students performed significantly better on end of class evaluations than students who worked on similar assignments on their
own [Rick et al., 2002].

2 Project Plan
Our overall project plan is:

• To create prototype authoring tools that ease the creation of learning suites for contextualized computing classes.

• To explore the benefit of these learning suites by using these tools to create alternative segments of our existing contextualized computing courses. For example, during a two weeks period, we might offer students a choice between a unit on advanced audio processing, or on MIDI, or on Excel, where all three units might lead students to the same learning objectives but in different contexts. During the alternative segment period, there would be reduced lectures for students, e.g., if a course typically has three lectures a week, only one lecture each week would be made available to students studying a given alternative segment.

• To evaluate (a) the process of authoring; (b) the costs of teaching alternative contexts supported by a learning suite; and (c) the impact on student learning and retention.

• To disseminate the tools and packaged segments, so that other schools can both create segments and use packaged segments. We will also create an on-line repository (based on CoWeb) for faculty to find and share segments, drawing on lessons from learning objects digital libraries [Laleuf and Spalter, 2001].

• To extend our evaluation to adopting schools, in order to study (a) the process of tailoring; (b) the costs of teaching with tailored (not self-authored) learning suites; and (c) the impact on student learning and retention.

• To disseminate our findings (a) to computer science faculty, for use in their classes and to improve the quality of computing education and (b) to the learning sciences community, to offer lessons learned on authoring, teaching with, and learning with learning suites. Our hope is that the design guidelines and experiences might be used to support contextualized learning suites in other disciplines.

2.1 Prototype Development Plan
Our plan is to develop a prototype authoring tool for creating learning suites for contextualized computing education that (a) ease the development of learning suites like those in current use at Georgia Tech and (b) facilitate better learning opportunities by extending our current model.

Supporting current use. Our current learning suites are unusual in that we have created everything associated with them, including the textbooks [Smith, 2005] [Guzdial and Ericson, 2005] [Guzdial and Ericson, 2005], as well as the lecture slides, homework assignments, and creating the spaces within the CoWeb for process guidance, anchored collaboration (e.g., for homework assignments and exam reviews), and galleries and cases.

The first role for our prototype authoring tool is to support and package current practice. We want to provide the rich collection of resources, and contemporaneous process support, in such a way that the package of resources and supports can be activated within and tailored for a given school’s context and calendar:

• We plan to support definition of a detailed process for the semester, going beyond simply a lecture and homework schedule, to define when process guidance should be updated. The

7
packaged version of a learning suite, when activated, would automatically update process guidance, create anchored collaboration spaces, and provide appropriate support at the appropriate point in the process.

- We plan to support the definition of all the interlinking resources, including lecture slides, homework assignments, exams, and exam reviews. Currently, our faculty-authors use traditional technologies for producing materials (e.g., HTML, Microsoft PowerPoint and Word, \LaTeX). To provide leverage on creation of so much material, we plan to support markup languages beyond HTML, such as Con\TeX which allow creation of printed material and electronic documents, including lecture slides, from a single content source\textsuperscript{6}.

The idea is to make sure that the appropriate learning resources and activities are easily generated by the author and are made available at the appropriate time (and recommended to the students at the appropriate point in the process). For example, in Week 3 of a given course, the Week 3 comments page should be linked to the process guidance “Hotspots,” and the Week 3 homework assignment questions page should be generated and added to the process guidance. If there is a relevant case library selection or gallery for the given homework assignment, it should also be added to the process guidance.

**Extending current use.** While our current use is built upon findings in learning sciences research, that same research suggests additional supports that we might provide to the author-teacher and to the student.

- *Explicit identification of learning goals.* We will invite users to be explicit about the learning objectives for their course, as was done in IByD (Instruction By Design) [Urdan et al., 1992] [Urdan et al., 1993]. The advantage for the author-teacher is that assessment can be conceptually linked directly from lectures and assignments and back to learning objectives—the teacher knows that the concept or skill has been presented, utilized by the student, and then tested. The advantage for the student (as suggested by studies of IByD [Urdan et al., 1992] [Urdan et al., 1993]) is that the educational activity is more explicitly designed to meet objectives.

- *Scaffolding for the author-teacher.* There are better and worse ways to write texts for students. Recent work at the University of Michigan has shown that well designed texts can themselves be supportive of science inquiry, even without any explicit additional technologies [Hapgood et al., 2004]. Especially given explicit identification of learning goals, our tool should be able to offer suggestions and recommendations to teachers on how to write learning resources in a way that encourages exploration and learning.

- *Monitoring tools.* Currently, teachers of a contextualized computing course simply visits the CoWeb space on a regular basis to monitor student progress and questions. If a teacher were to be monitoring several contextualized courses or segments of courses, simply “looking around” will be costly in terms of time and inefficient. We propose creating monitoring tools so that teachers can easily see activity in the CoWebs for each segment and respond to students efficiently.

- *Take advantage of programming-as-text.* One of the advantages of computing for learning suites is that the “stuff” of interest of computing is already digitized and easily manipulable, e.g., program texts. While there have been efforts to make program texts easier to explore and more like literature for students [Knuth, 1992] [Knuth, 1986], few of these have been used in computer science classrooms because of the difficulties of authoring programs for such use [Shum and Cook, 1994]. By controlling the authoring tools, and directly linking them into the learning suite, we can make the production of literate programs easier for the author-teacher.

\textsuperscript{6}http://www.pragma-ade.com/
and more directly useful for the student, e.g., by linking to the appropriate points of a program directly from an on-line text or lecture slide, and vice-versa.

- **Linking to additional resources.** We plan to provide simple linking to additional resources for computing education, such as algorithm animations and software visualizations [Naps et al., 2003]. As our strategy for the tools is to emphasize automatic generation of language markup (see below), we plan to explore the use of SMIL [Bulterman and Rutledge, 2004] to easily create visualizations and compelling presentations for learning resources.

**Development and Usability Strategies.** How do we make it possible to do all of this, and be usable, and still produce prototypes and not engage in full-scale development? We plan to make extensive use of **generated language markup**. Notations such as HTML, \LaTeX ConTeX, SMIL, and CWEB are markups (inserted tags) into normal text. There are tools available now and becoming available that provide highly usable interfaces with facilities for generating marked up text, with customizations to generate new kinds of markup.

We are considering two possible tools for our development.

- **Frontier.** Frontier is an end-user programmable database that features outline and plain text entries, with an easy-to-use interface for both Macintosh and Windows platforms [Neuburg, 1998]. Frontier is notable for being the first blogging tool. Users of Frontier created a set of scripts for generating HTML from outline and text views, and thus created easily extensible Web pages without the user having to know HTML [Cadenhead, 2004]. We can use these same technologies and interfaces to retain the ease of text entry and manipulation, but create other markup languages to meet our purposes. We have had experience in using Frontier for similar cross-media purposes, e.g., using Frontier to enable desktop access of resources from a MOO [Guzdial, 1997].

![Figure 4: A screenshot from Frontier showing outline, plain text, and database views](image)

- **Squeak.** Squeak is a flexible and powerful cross-platform development environment that is particularly strong at multimedia and Web development. The Collaborative Software Lab that the PI directs is uniquely skilled at Squeak development [Guzdial, 2001a] [Guzdial and Rose, 2001]. The CoWeb tool is already developed in Squeak [Guzdial et al., 2000], so there would be some leverage in terms of creating links to our collaboration spaces by developing in Squeak.

9


2.2 Evaluation Plan: Settings and Questions

We are planning on two kinds of evaluation settings for this project. In both settings, we plan to study both author-teachers and students.

- **Segments within existing classes.** Our first trials with our learning suites authoring tool will be in our existing contextualized computing courses. Our plan is to create parallel units, segments within an existing class, each of which aims to achieve the same learning objectives, but each contextualized differently. For example, we could imagine introducing iteration concepts using sampled sound examples and assignments in our media course (as we do now), or using MIDI (which is actually more relevant and interesting to music majors), or using Excel (which is more relevant and interesting to management majors). We could literally allow. Similarly, we could create segments that introduce C concepts in our engineering CS1.

Rather than having additional faculty, our plan is to use the same teacher for the course, but to split up the number of lectures available during the parallel segments among the segments. If there are three lectures per week, three parallel segments were being introduced, and the segments were to last for two weeks, we would have one lecture per week for students. This reduction of lecture time reduces the cost for the teacher (and the department), but moves the course closer to an online model and all of the correlated problems with retention and learning [Bernard et al., 2004].

- **Adoption by other teachers.** We anticipate that some of our segments, especially those that following the existing contextualized courses that are being adopted by others [Smith, 2005] [Guzdial, 2004] [Guzdial and Ericson, 2005], will be used by other teachers when made available. The advantage for these (typically, external to Georgia Tech) teachers will be that utilizing the ready made segments will set up all of the learning suite infrastructure (e.g., CoWeb with galleries, Hotspots, and other supports) that we have found useful in contextualized computing courses.

Our evaluation plan thus has three sets of participants:

- **Author-Teachers.** We want to know how effectively the tools support the author-teachers. How hard is it to develop learning suites with these tools? Are there supports (e.g., other kinds of automatically generated linkages) that are missing that the teacher would like to use? How hard is it to teach with these tools? Are there supports (e.g., other monitoring tools) that are missing that would make it easier to monitor and respond to students need when teaching with a segment?

For these participants, we will ask them to keep a log of issues as they arise during their authoring and teaching. We will additionally use logging in the authoring tool to be able to compute time required to create a learning segment. We will use surveys, with possible follow-on interviews if the participant consents, to explore the usefulness and value of the authoring tools and segments.

- **Adopting Teachers.** For these participants, the teaching questions are the same as for the author-teachers, but we also want to know about how the adopting teachers tailor the produced learning suite segments for their students. How hard is it to tailor? How effective were the tools in supporting the tailoring in the ways that the teacher wanted to tailor the segments. We will similarly ask these participants to keep logs, to allow us access to their tool logs, and to complete surveys (and possibly interviews).

- **Students.** Students who learn with the learning segments are of particular interest to us. We can only offer multiple contextualized classes with the same number of faculty, if there is less contact time between student and faculty—yet we know that the closer we get to “on-line”
learning, the less learning and retention occur [Bernard et al., 2004]. We plan to use surveys after segments to ask students about their motivation and how much they enjoyed the contextualized segments. With their permission, we plan to compare performance on examinations (on isomorphic problems, as we did in [Guzdial and Kehoe, 1998]) and to compare retention with the same classes in previous terms that did not offer additional contextualized parallel segments. Is retention similar? Is learning similar? Do we get more motivated students due to the increased contextualization?

2.3 Dissemination Plans

We have two sets of artifacts and findings to disseminate in this project. The first is the learning suite authoring tools and the packaged learning suite segments produced through use of these tools. The second are the results of our studies of the author-teachers and the students.

Our plan is to make the tools and segments available freely on our project website\(^7\). We will advertise these tools and authored segments on the ACM SIGCSE (Computer Science Education) members list and in our SIGCSE Conference workshops (see below). The PI already publishes frequently in the computer science education community: In ACM SIGCSE (Computer Science Education) conference, the international ITICSE (Innovation and Technology in Computer Science Education) conference, and the IEEE/ASEE FIE (Frontiers in Education) conference. We will continue to publish our work, both about the artifacts and findings in these forums as part of this project. We expect to submit one-to-two papers per year in some combination of these CS Education conferences. We also plan to submit a paper and materials to the new ACM Journal of Educational Resources in Computing\(^8\) for review of the material and dissemination at a broadly accessible (and indexed by the ACM Digital Library) level.

We plan to offer ACM SIGCSE Conference workshops in Years 2 and 3 of the proposed work, to teach others to use the authoring tools and to tailor segments. For the last three years, we have offered a workshop on *Multimedia Construction Projects in CS1 and CS2* that has met with success (e.g., filled enrollments with evaluations above the average for SIGCSE workshops). We plan to offer a similar workshop on authoring learning suites for contextualized computing, drawing on what the learning science community knows about the best way to teach teachers in workshops [Kolodner et al., 2003] [Linn and Hsi, 2000]. Through this workshop (as well as through contacts with faculty who seek to adopt our contextualized approaches), we hope to find volunteer *adopting teachers* for our study.

Our results will be particularly interesting to the learning sciences community. We believe that our findings will be relevant to those in science and mathematics education who would like to explore similarly contextualized education. We plan to submit at least one paper over the three year project plan in each of Computer Supported Collaborative Learning Conference and International Conference of the Learning Sciences. What we learn about computing education will be relevant to the Computing Education Research journal and to the ACM-sponsored International Computing Education Research Workshop—we anticipate at least one paper to the workshop and to the journal in the three year period.

3 Conclusion and Project Schedule

This is a high-risk proposal. The history of technology development by teachers is riddled with failed attempts [Solomon, 1986] [Cuban, 2003]. Introducing reform into higher education is particularly challenging with depressingly little success [Cuban, 1999]. However, the setting of this proposal is almost a best-possible scenario. The curriculum reforms described are already in place at a high-visibility institution, and they are already being adopted by other institutions. The PI has a track record of producing usable multimedia authoring and collaboration technologies that teachers do

\(^7\)We have already established a website for adopters of our media computation approach at [http://coweb.cc.gatech.edu/mediacomp-teach](http://coweb.cc.gatech.edu/mediacomp-teach).

\(^8\)[http://www.acm.org/pubs/jeric/homepage.html](http://www.acm.org/pubs/jeric/homepage.html)
adopt and invent with (e.g., MediaText [Hay et al., 1994] and CoWeb/Swiki [Guzdial et al., 2001b]). It’s a worthwhile gamble to make contextualized computing education more economical to adopt and more likely to succeed.

Our overall proposed project schedule follows:

- **Year One** (assuming a January 2006 start date)
  - *Winter 2006*. Produce our first prototype authoring tool.
  - *Summer 2006*. Author the first set of parallel segments for one of our contextualized CS1 courses (probably our media computation CS1).
  - *Fall 2006*. Use the parallel segments structure in a CS1 and conduct the first student evaluation. We will also submit the SIGCSE Workshop for presentation in February 2007.

- **Year Two**
  - *Winter 2007*. We will analyze the results of the student and author evaluations. If time allows, we will quickly modify the segments based on the first year evaluations, and conduct a second trial of the segments. We will also present our first SIGCSE workshop.
  - *Summer 2007*. We will prepare additional segments at Georgia Tech for exploration in our contextualized courses. Working with faculty who show interest during the first SIGCSE workshop or who contact us about adopting our materials, we plan to begin studying adopting teachers.
  - *Fall 2007*. We will run a third study in Georgia Tech classes, while conducting the first class studies of tailored segments at some external schools. We will also submit the second SIGCSE workshop.

- **Year Three**
  - *Winter 2008*. We will focus on analysis of Year Two results. Based on Years One and Two results, we will iterate on the authoring tool. We will conduct another SIGCSE workshop with our improved tools and all segments authored thus-far.
  - *Summer 2008*. We will continue studying the authors at all schools.
  - *Fall 2008*. Our focus at this point will be gathering and analyzing data from all uses of the authoring tool and segments, and preparing final papers on our results.
4 Budget Justification

Our budget requests funding for the PI and one graduate student research assistant. With the PI, the graduate student will develop the prototype authoring tool. The graduate student and PI will work together on survey instruments and interview guides. The graduate student will be primarily responsible for data collection. The graduate student and PI will be jointly responsible for data analysis and results dissemination. The PI will be responsible for workshop development and presentation.

- **Personal Services**
  - Guzdial is funded at one summer month per year.
  - We will share 0.25 graduate student research assistant (GSRA) with other labs for hardware support.
  - We will hire one GSRA for development and evaluation.

- **Other**
  - We are asking for travel funding to attend conferences for dissemination.
  - We are requesting M&S funding for software and other costs.
  - We are also requesting funding for GSRA tuition and charges for computing infrastructure support.
5 Facilities

The College of Computing maintains a variety of computer systems in support of academic and research activities. These include more than 50 Sun, Silicon Graphics, and Intel systems used as file and compute servers, many of which are quad-processor machines. In addition, there are more than 1,000 workstation class machines from Sun, Silicon Graphics, Intel, and Apple especially for student use. A number of specialized facilities augment these general-purpose computing capabilities. The hardware that will be purchased for this project will be of similar quality to what the students use, for testing purposes, but will be set up to facilitate development.

The Graphics, Visualization, and Usability (GVU) Center houses a variety of graphics and multimedia equipment, including high-performance systems from Silicon Graphics, Sun, Intel, and Apple. The affiliated Multimedia, Computer Animation, Audio/Video Production, Usability/Human Computer Interface, Virtual Reality/Environments, Electronic Learning Communities, Computational Perception, Software Visualization, Biomedical Imaging, Collaborative Software, and Future Computing Environments labs provide shared facilities targeting specific research areas. These laboratories’ equipments will be of use in developing our multimedia projects.

PI Guzdial is the Director of the Collaborative Software Lab, affiliated with GVU. The Collaborative Software Lab has a bank of ten servers supporting our experimental software for studying computer-supported collaborative learning. In addition, we have three Linux workstations, two NT workstations, and two Apple workstations used for development. The focus of the Collaborative Software Lab is on facilitating multimedia collaboration, so multimedia facilities available include a high-end Alesis keyboard, projection facilities, a Canon digital video camera, and a Sony Mavica digital camera.

All of the College’s facilities are linked via local area networks which provide a choice of communications capabilities from 10 to 1000 Mbps. The College’s network employs a high-performance OC12C (622 Mbps) ATM and GigabitEthernet (1000 Mbps) backbone, with connectivity to the campus ATM network provided via OC12C. The primary campus Internet connection is provided by a direct 100 Mbps link to the service provider’s Atlanta switching center, augmented by OC3C ATM and OC12C connections, respectively, to the NSF vBNS (very high performance Backbone Network Service) and Abilene research networks. Georgia Tech is also leading southern regional gigabit network efforts (SoX.net, the Southern Crossroads) as part of Internet2.

Additional computing facilities are provided to the Georgia Tech campus by the Institute’s Office of Information Technology (OIT), including five public-access clusters of Sun, Apple, and Dell workstations, a collection of Sun multi-processors which are treated as a single computational resource via login load sharing, and various mainframes.
References


