Whittier, D. (2005). The teacher as software developer. *Contemporary Issues in Technology and Teacher Education*, *5*(1), 66-72.

The Teacher as Software Developer

David Whittier
Boston University

Abstract

The Teacher as Software Developer is the name of a program integrating technology instruction, curriculum, and field experiences in teacher preparation. In an introductory education course for all undergraduate education students, a required technology lab links to a one-day-a-week prepracticum. Preservice students produce a Web site or “software” for their supervising teacher, who is their "client,” and the supervising classroom teacher directs them to authentic curriculum objectives. Preservice teachers learn about software while learning about teaching, lesson planning, curriculum, and technology. Supervising classroom teachers’ gain an opportunity to experience designing and using software with their own students and curriculum and access to the preservice student produced Web sites after the student has completed the lab. Evidence of the program’s success comes from student survey data and student reflections. Despite the program’s productivity, it remains an island of instructional technology.

The Teacher as Software Developer is the name of an innovative program synthesizing technology instruction, curriculum content, and field experiences in teacher preparation. It stands for the organizational framework of an introduction to technology lab required of all undergraduate education majors at Boston University. Since introducing this program in 1998, students have benefited from the transformation of technology instruction at our School of Education. Prior to its introduction, there were, at best, good faith attempts at isolated versions of technology instruction. Technology instruction for the inexperienced preservice undergraduate was a largely futile exercise in *imagining* what software would look like in a classroom. At worst, it was an exercise in putting technology mastery first, and leaving content to chance. Through The Teacher as Software Developer program, technology instruction has become an agent and a catalyst for our undergraduate preservice teachers learning about lesson planning, teaching, curriculum, and technology. This article provides a description of the Teacher as Software Developer program and how it fits into the undergraduate curriculum for preparing teachers.

The Teacher as Software Developer program describes both the arrangement of project-based learning in our introduction to technology lab and the planned outcome of the lab, other coursework, and field experiences that comprise the teacher preparation program. That is, preservice teachers who successfully complete the introduction to technology lab are well on their way to becoming teachers who can produce instructional software customized to their own classroom. The program does not refer to teachers writing code for Microsoft or some other computer-oriented activity. It does refer to teachers who are capable of developing their own instructional Web sites and other multimedia resources using a variety of authoring instruments available today and who are comfortable with their students doing the same.

The Teacher as Software Developer program describes the conceptual framework of a technology lab that is integral to the fundamental education course required of all undergraduate education students. SED ED 100, Introduction to Education, is a major six-credit course that education students typically take in their first or second year at Boston University, well before they take methods courses, which they take typically in their third year. ED 100 orients students to the teaching profession and includes critical, introductory instruction in "teacher competency, philosophical concepts applied to education, schooling as a societal system in itself, including character education, curriculum, administrative organization, learning principles, the emergence of the educator in society, and the education of special populations in schooling" (Boston University Undergraduate Bulletin, 2004-05). Students taking ED-100 are also required to take concurrently SED ED 101, our introduction to educational technology lab, and to participate in a one-day per week field placement in the classroom of a local school.

The program was established in 1998, after several semesters of unsuccessfully attempting to link the introduction to technology directly with the broad introduction to the teaching profession course. The transformation occurred through linking the introduction to technology lab to the prepracticum field placement required of each student. This arrangement, referred to as the Teacher as Software Developer program, has proven vastly more productive in helping students learn about teaching and technology. Evidence for this conclusion comes from observing the projects the students produced. Before the Teacher as Software Developer program, students were either addressing broad philosophical and cultural topics from the lecture course that were too difficult to be realized in the limited context of the lab, or they were addressing some imagined application in the classroom for which they had insufficient experience to envision. Because of the difficulty of achieving these objectives, the content of the student projects tended to drift, taking a back seat to technical competency. Given these content obstacles, far too many projects ended up as "travelogues," rather than educational software. Although students were able to demonstrate sufficient technical mastery, the projects were not well grounded in realistic classroom applications. Linking to the prepracticum overcame these obstacles.

The prepracticum placement assigns students to be in a classroom one full day a week. The Teacher as Software Developer program builds upon this fieldwork in the introductory technology lab by requiring that assignment work done in the lab serve the prepracticum classroom. In the lab, students learn basic technology and curriculum integration skills, and because they are producing resources for use in the classroom, they are in the position of being a "Teacher as Software Developer." Based on a simple agreement between the preservice teacher and the supervising classroom teacher, the design builds on the premises of project-based learning (Moursund, 1999). The lab counts as 10% of a student's grade in the six -credit ED-100 course, which although considerably less grade value than the level of work required in the lab that meets one hour per week for the entire semester, is an improvement over the zero credit status it carried until fall of 2003.

The technology resources taught in the lab have changed over the years as technology has evolved. In the current form, preservice students agree to produce a Web site for their supervising teachers using Dreamweaver. The Web site must include both instructional and assessment components (which are what we refer to as "software") for technology-infused lessons. The Web was chosen because of its easy accessibility and revise-ability, which overcome two major historical obstacles teachers have faced in using technology-based resources. Dreamweaver was chosen because of its currency as Web-authoring software.

To get started producing their Web sites, the preservice students are instructed to think of their supervising teachers as their "clients" for this lesson/software. The supervising classroom teachers agree to see that the lesson addresses carefully selected, authentic curriculum objectives at the appropriate time, approximately 8-10 weeks into the semester. Even though the agreement itself is simple and clear, only careful targeting and specific support ensures that these experiences are productive.

While the purpose of the program is to support the learning of the preservice teacher, anecdotal evidence suggests that benefits also accrue to the supervising classroom teacher. These benefits appear to occur through providing the in-service classroom teachers experience in designing and using software with their students in their particular classrooms and focused on their particular curriculum objectives without having to take the time to produce it themselves. Through accepting responsibility for directing their preservice teacher to appropriate objectives, the supervising teachers gain the assistance of energetic undergraduates who bring the supportive environment of the university-based instructional technology lab to the agreement. Part of the explanation for this benefit may be that most teachers did not learn to prepare technology -infused curriculum resources in their teacher preparation programs. It is difficult to have time to learn *and* time to construct their own instructional software while being responsible for their classroom, let alone have the resources of the university to draw upon. The Teacher as Software Developer program aims to prepare teachers to produce these types of resources *before* they have responsibility for a classroom. It also works toward preparing future teachers to work in partnership with instructional technology specialists, much in the way they are working with classroom teachers in the program.

By taking seriously the task of directing the preservice teachers to appropriate curriculum objectives, participating teachers have the opportunity to gain valuable experience in using software as a piece of their instruction. Additionally, supervising classroom teachers' may benefit from being able to continue to use the student produced Web sites after the preservice teachers have completed the technology lab. Many teachers have reported that these resources are valuable and have requested continued access to them. In response, we are building a searchable Web database of URLs, and we are looking for support to collect data on how supervising teachers are continuing to use their student-produced Web sites.

Putting teachers in the driver's seat of software design forms a theoretical framework for the program by overcoming two primary causes for the failure of software and technology-based materials to support learning. The first of these problems results from overemphasizing software as a stand-alone product not sufficiently integrated into the processes and methods that teachers employ to help students learn. Historical analysis of many failed attempts to include technology in the classroom shows that technology materials designed by nonteachers overly emphasize generalized content transmission instead of specific learning by individual students (Saettler, 1990). Emphasizing products over process describes the emphasis on products comprised of generalized content produced by nonteachers, as opposed to products that support and fit closely with classroom processes. Emphasizing products that support process means to support specific learning by individuals who have strengths and weaknesses and who need to interact with new ideas in order to assimilate them, which is the kind of thing teachers do routinely. The result of emphasizing products without regard to process is ultimately the low productivity of technology in the classroom (Cuban 1986, 2001; Dockterman 1988). The Teacher as Software Developer program avoids the pitfall of putting products over process by making teachers the principle architects of software. This puts teachers in charge of ensuring that the software complements their teaching style, is carefully adapted to the needs of their particular students, is focused on the goals of their curriculum, and is carefully integrated into the flow of learning activities in the classroom.

The second common cause for the failure of software to support learning is poor alignment between the educational objectives and content of the software and the curriculum objectives of that individual teacher and his or her particular class of students. Even with state curriculum standards, teachers know that their instructional strategies will work best when they match their own strengths to the strengths and weaknesses of their students. It is very difficult for nonteachers, unfamiliar with the great variety of students and teaching styles common in schools, to design and produce software that will meet the needs of an individual teacher teaching in an individual school. This is the lesson that comes through so powerfully in the research of Cuban (1986, 2001) and the historical analysis of Saettler (1990). The Teacher as Software Developer program addresses this weakness by allowing teachers to direct the development of software for their students, while consuming very little precious time. At the same time, the program gives a preservice teacher an early opportunity to produce, implement, and evaluate software in support of curriculum and instruction and, hence, be on their way to becoming a teacher as software developer. Molding software to fit perfectly into the flow of instruction in the classroom can then be the foundation for the habit of turning to technology as a lively and flexible teaching method.

Another key point about the Teacher as Software Developer program is that, because this is an introductory course, students have four years to learn about integrating technology and their teaching. Of course, to take advantage of that window of opportunity, there would need to be further, systematic instruction in the design, development, and evaluation of educational software. Plans at Boston University include the introduction of a required lab on universal design and assistive technology that would follow the introductory lab, as well as a lab attached to methods courses and a requirement for utilizing technology during student teaching. These labs and requirements would comprise a comprehensive and systematic approach to preparing teachers to use technology. Unfortunately, the four years are currently unrealized potential as to date; there are no formal requirements for further instruction and practice in utilizing technology beyond the introduction to technology lab. Some faculty members require technology-infused assignments and others do not, meaning that continued development of curriculum and technology integration competencies are not systematic.

Even though the value of the Teacher as Software Developer program is focused on learning to make instructional software, its benefits go beyond learning computer competencies. By helping preservice teachers learn about appropriate computer software in the context of learning about essential teacher concerns such as curriculum, lesson planning, and assessment, technology becomes an agent for learning about teaching. In this way, the program not only educates prospective teachers but also models the integration of teaching, curriculum, assessment, and technology.

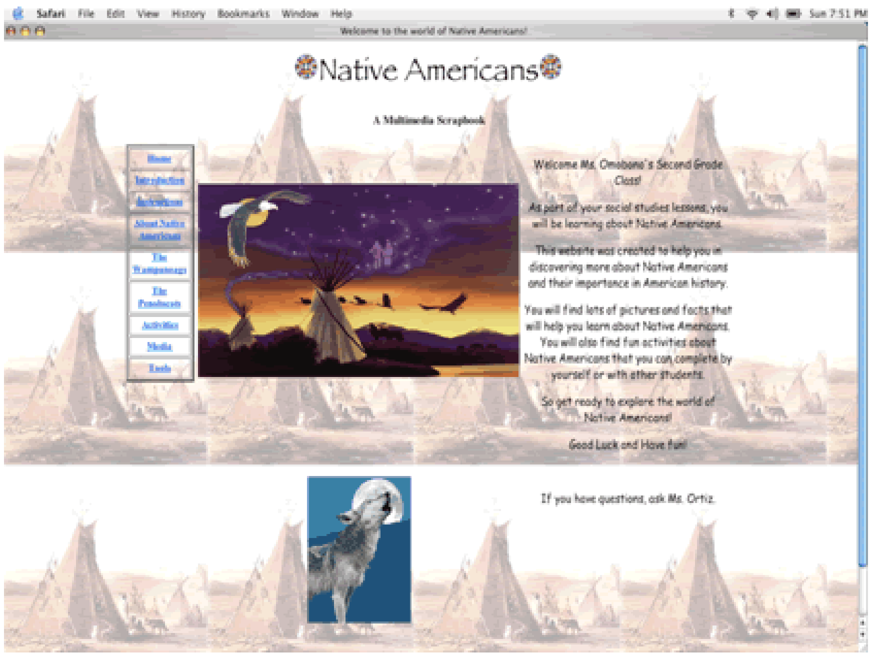
Evidence of the success of the Teacher as Software Developer program comes from student self-assessed survey data and from student reflections on their experiences in their classroom placements. For example, one freshman wrote, "The technology lesson I did with Mrs. W's first grade class was probably one of the best experiences of my life. I learned many things about myself ... because it showed me that I have what it takes to be a decent teacher" (Student Paper, 1999). Another wrote,

> This past Wednesday was the day I presented the [KidPix] slideshow to the students on the 'moose' (computer attached to a TV). I have never seen anything hold their attention for as long of a time. They waited in anticipation for their slide to come up. After viewing the slide show once, we had each child come up and read his/her poem to the class. It was wonderful to see the pride on their faces as they read their poems. (Student Paper, 1999)

More excerpts of student reflections are accessible on our Web site at http://emt.bu.edu/TasSD.

Cooperating teachers at the Alcott Elementary School in Concord, Massachusetts helped Boston University faculty test and refine the Teacher as Software Developer program in the context of a Massachusetts Department of Education Technology Literacy Challenge Grant (1998-2000). Preservice teacher participants completed surveys during the fall 1999 and spring 2000 semesters of the program’s implementation to determine its impact upon the preservice teacher’s competencies. An instrument developed at Boston University based, in part, on International Society for Technology in Education (2002) National Educational Technology Standards for Teachers and, in part, on a similar self-assessment used in 1999 in the Boston Public Schools (Boston Public Schools, n.d.) measured the preservice teacher’s self-assessment of their technology skills (Part 1) and what they knew about technology and curriculum integration (Part 2). Results of “Part 2: Supporting Teaching and Learning With Technology,” which focused on technology and curriculum integration, showed that participating School of Education student’s reported an average growth of over 38% (n = 49) from their precourse self-assessment to their postcourse assessment.

Participants’ preprogram scores on Part 2 ranged from 0% to 85% (median 27%, n = 28) and postprogram scores ranged from 23% to 100% (median 69%, n = 21). No data were collected on these students after the completion of the program. The survey instrument is available through the research section of the Teacher as Software Developer Web site.



Welcome to the world of Native Americans!

Native Americans

A Multimedia Scrapbook

Welcome Ms. Omobono's Second Grade Class!

As part of your social studies lessons, you will be learning about Native Americans.

This website was created to help you in discovering more about Native Americans and their importance in American history.

You will find lots of pictures and facts that will help you learn about Native Americans. You will also find fun activities about Native Americans that you can complete by yourself or with other students.

So get ready to explore the world of Native Americans!

Good Luck and Have fun!

If you have questions, ask Ms. Ortiz.

Figure 1. Second Grade New England Native Americans Home Page

The success of this program derives not only from successfully integrating the talents of student teachers with the wisdom of supervising teachers, or from its assuring that the substance of the learning must always prevail over the dazzle of the technology, but also from steadfast attention to the nuances of each lesson. Without this last ingredient, the Teacher as Software Developer program would not be so highly regarded by our students, university colleagues, and cooperating teachers. The Teacher as Software Developer program has shown how all teachers, young and old, can learn the most effective and most appropriate use of technology. Examples of preservice teacher produced instructional Web sites include a site supporting a second grade classroom social studies curriculum on New England Native Americans (http://ed101.bu.edu/studentDoc/Fall04/nikki06/index.html; see Figure 1) and a site of "personalized reading activities" for a second grade class (http://ed101.bu.edu/studentDoc/Spring04/gilberts/site/index.html).

Despite the productivity of the Teacher as Software Developer program, it remains an island of instructional technology, as there are currently no other required technology courses or labs in the four-year undergraduate experience of training to become a teacher. Most methods faculty members have introduced technology-infused assignments, in many cases stimulated by participating in the Boston University PT3 grant project focusing on faculty development (Whittier & Lara, 2003), but there is no further systematic education in effectively using technology in teaching. In addition, the use of technology is not required in the fourth year student teaching experience. This may also further erode success at effectively integrating technology into teaching (Strudler, McKinney, Jones, & Quinn 1999).

Although the Teacher as Software Developer program has proven productive in helping preservice teachers to have an early experience of effectively using technology in support of instructio n and in providing cooperating teachers with instructional resources both during its production and after the software producer/preservice teacher has moved on, its long-term impact is yet unknown.

References

Cuban, Larry (1986). *Teachers and machines: The classroom use of technology since 1920*. New York: Teachers College Press, Columbia University.

Cuban, L., (2001). *Oversold and underused: Computers in the classroom.* Cambridge, MA: Harvard University Press.

Docktorman, D., (1988). *Tools for teachers: An historical analysis of classroom technology.* Unpublished doctoral dissertation, Harvard University.

Boston Public Schools. (n.d.). *LINC Boston technology plan*. Retrieved April 13, 2005, from http://boston.k12.ma.us/linc2/

Boston University Undergraduate Bulletin. (2004-05). Retrieved April 13, 2005, from http://www.bu.edu/bulletins/und/item18e3.html

Moursund, D., (1999). *Project-based learning using information technology* . Eugene, Oregon: International Society for Technology in Education.

Saettler, L. P., (1990). *The evolution of American educational technology.* Englewood, CO: Libraries Unlimited, Inc.

Strudler N., McKinney M., Jones, P., & Quinn, L. (1999). First-year teachers' use of technology: Preparation, expectations and realities. *Journal of Technology and Teacher Education, 7* (2) 115-129.

Whittier, D., & Lara, S. (2003). *Preparing Tomorrow's Teachers to Use Technology (PT3) at Boston University through faculty development.* Estudios Sobre Educacion, 005/2003.

Author Note:

David Whittier
Boston University
Email: whittier@bu.edu