Scaffolding active programming instruction with theoretically grounded screencasts and annotated worked examples
NWACC 2008-09 Proof of Concept Grant
Final Report – May 20, 2009

Laurie Murphy and David Wolff
{lmurphy, daw}@plu.edu
Department of Computer Science and Computer Engineering
Pacific Lutheran University, Tacoma, WA 98447
Phone: 253-535-7400 Project Web site: http://www.cs.plu.edu/~nwacc/

Project Goals:

Since moving to a new building with computer classroom facilities we have made a number of pedagogical improvements to our introductory programming course. These have included hands-on, in-class activities such as guided worked examples [1], where students follow along with us as we demonstrate the programming process, and pair programming [5] exercises. Not surprisingly, however, these hands-on activities make it challenging to cover as much material as we would in a lecture-based course. We have also come to realize that although students enjoy the hands-on activities, anecdotally, we have not seen commensurate improvements in students' grasp of the concepts they are designed to teach. We believe this is partly due to the cognitive requirements inherent in the hands-on exercises; students must listen and follow along with us, interact with the operating system to manage their files, cope with the editor commands and options, type their code correctly, debug any errors they introduce along way, and somehow manage to catch up if they fall behind. It is not surprising the concepts may get buried in all that clutter!

This project was designed to address our concerns over reduced content coverage and potential student cognitive overload. The pedagogical, assessment and dissemination goals outlined in our proposal include:

1) Creating priming presentations based on Mayer's research-based multimedia learning theory principles [2]
2) Creating guided worked examples [1] to illustrate the programming process
3) Making both the priming presentations and worked examples available to students as video podcasts from the class webpage and via RSS feed
4) Assessing the effectiveness of the video podcasts using an end-of-term questionnaire, a modified version of our web-based electronic classroom assessment system [3], and traditional summative assessments.
5) Sharing our results through papers or tutorials with CS education colleagues at, for example, the Northwest Consortium for Computing Sciences in Colleges (CCSC-NW) or ACM Special Interest Group on Computer Science Education (SIGCSE) conferences.

Results:

As of spring 2009, we have produced twelve video podcasts, each approximately 20 minutes in length. Rather than creating separate videos for the priming presentations and guided worked examples, we integrated elements of both into each podcast. Ten of the podcasts use priming presentations to review or introduce important programming concepts, and then guide students through a worked example that demonstrates the programming process from start to finish. Each of these podcasts is strategically disseminated a day or two before the associated lab session. The remaining two podcasts are made available at the beginning of the semester and guide students through installing required software on their own machines.

When we initially deployed the podcasts in fall 2008, we made seven videos available throughout the semester from our course management site and via RSS feed, which we encouraged students to subscribe to at the beginning of the term. An end-of-term survey revealed that fewer students than we had hoped actually watched the videos, with 23% of students watching none and only 31% of students watching three or more of the seven podcasts (see Figure 1). In the spring of 2009, we produced five additional videos, encouraged students to subscribe on multiple occasions, and linked to the podcast repository directly from the main course page. We also improved dissemination by linking each podcast directly from a particular lab assignment and recommending that students watch them in preparation for their lab sessions. Results were greatly improved with only 8% (2 students) watching none and over 62% of students watching at least 7 of the 12 available videos (see Figure 2). The percentage of students who reported the videos were very or somewhat useful also jumped from slightly less than 60% in the fall to over 90% in the spring.

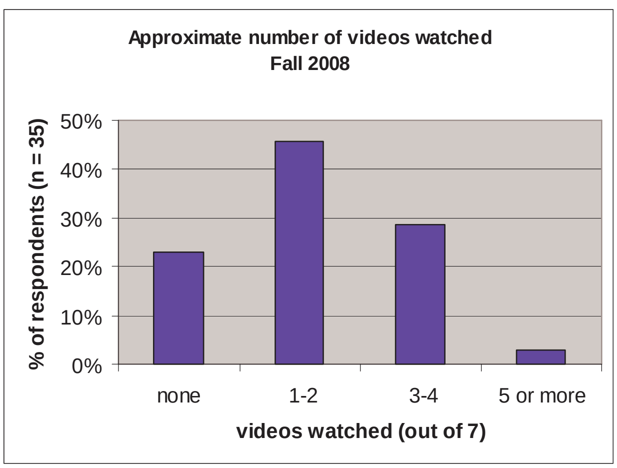

Figure 1

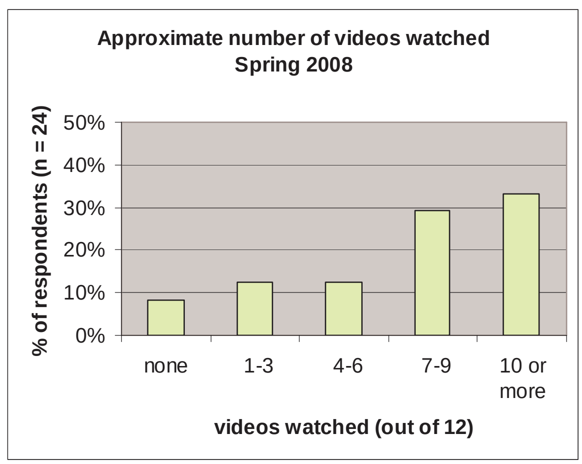

Figure 2

Students' reasons for watching the videos also changed from fall to spring, with nearly 90% indicating they watched the videos to prepare for lab and over 65% to gain a better understanding of the programming process in the spring (see figure 3).

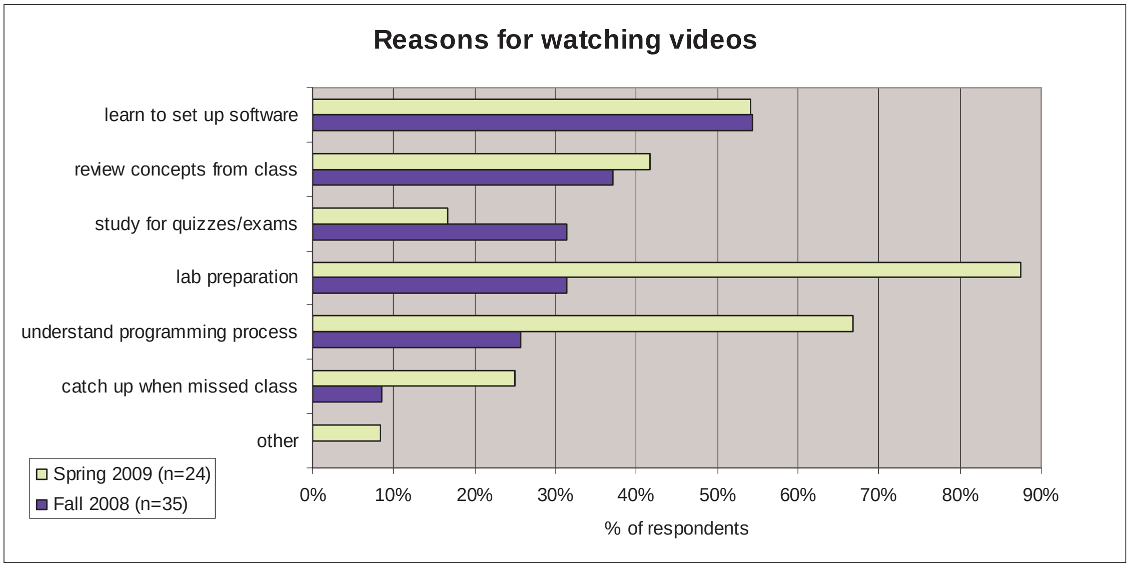

Figure 3

As part of the project we also updated our online classroom assessment system. It now allows a student email address field to be optional (previously it was included, even if students were not required to fill it in), permits a variable number of questions (previously there were always two), and incorporates the selection of multiple preset question sets. We continued to use this system as a means of getting feedback at the end of lab sessions throughout the year, but still need to work on how best to integrate its use in the classroom.

Impact/Future Plans:

Along with the positive impact the survey results suggest for programming instruction, anecdotally we noticed that the occasional hyper-frustrated (sometimes to the point of tears) students all but disappeared from the labs during spring term. We also did not hear the commonly uttered complaint that students didn't even know where to begin on their labs. An unexpected phenomenon that may account for these improvements was that students often referred to the videos (typically without sound) as a source of examples as they programmed during lab sessions. In fact, the number of students who reported watching the videos on the lab/classroom computers jumped from 14% in the fall to 42% in the spring. Our lab grader also commented that she noticed more students than in previous semesters submitted labs that received 'B' or 'C' grades, as opposed to receiving failing grades, or not submitting the labs at all. This suggests the videos may provide struggling novice programmers with much needed scaffolding, although more data is needed to draw any definitive conclusions, particularly regarding learning outcomes.

We also feel the project has much to offer other educators who are considering producing podcasts for their courses. We are pleased to report that a paper describing the lessons we learned about how to effectively encourage students to view the video podcasts [4] was accepted at the 2009 CCSC-NW conference. We will present the paper at the conference here at PLU in October, and it will appear in the December issue of the *Journal of Computing Sciences in Colleges (JSCS)*.

Future plans include additional quantitative analysis of our spring survey, a qualitative analysis of written course evaluation comments and student responses submitted via our online assessment system. We also hope to compare overall student performance in semesters when the videos have and have not been used. We plan to submit a paper describing our results to the 2010 SIGCSE Symposium.

We will continue to use the podcasts in our introductory Java course, making revisions and modifications as necessary. We would also like to repeat our survey, but this time include demographic data to see if there are differences in viewing habits or impressions of the videos with respect to gender or age. We have also recently made the videos available to educators and students outside of PLU from our NWACC project web site.

Related Web Sites:

Project web site: http://www.cs.plu.edu/~nwacc/
Course web site: http://www.cs.plu.edu/courses/csce144/spring09/

Publicity:

The grant was reported to the PLU community and Board of Regents. All of the videos are now available from the project web site, and a link will be included in the paper [4] we will present at CCSC-NW. The paper will also appear in the CCSC-NW online program (http://www.ccsc.org/northwest/2009/), and in the *JCSC*, which is accessible through the ACM's Digital Library (http://portal.acm.org/portal.cfm).

Final Budget:

		budgeted	spent
Faculty summer stipends:		$6,000.00	$6,000.00
Student web developer:		$400.00	$400.00
Hardware & Software:		$2600.00	
	Camtasia Screen recording software (PC – 2 copies)		$311.17
	ScreenFlow Screen recording software (Mac)		$84.99
	Survey Monkey subscription (for data collection and analysis)		$200.00
	Microphone		$107.95
	Tablet PC		$1,996.91
Travel:		$1,000.00	
	SIGCSE 2009		$898.98
Total:		$10,000.00	$10,000.00

References:

[1] R. Atkinson, S. Derry, A. Renkl and D. Wortham, Learning from Examples: Instructional Principles from the Worked Examples Research, *Review of Educational Research*, vol. 70, 181-214, 2000.

[2] R. Mayer. *Multimedia Learning*. Cambridge University Press, 2001.

[3] L. Murphy and D. Wolff. Take a minute to complete the loop: using electronic Classroom Assessment Techniques in computer science labs. *Journal of Computing Sciences in Colleges*. 21:1, 150-159, 2005.

[4] L. Murphy and D. Wolff. Creating video podcasts for CS1: Lessons learned. Forthcoming in the *Journal of Computing Sciences in Colleges*, 2009.

[5] L. Werner, B, Hanks, and C. McDowell. Pair-Programming Helps Female Computer Science Students. *ACM Journal of Educational Resources in Computing*, Vol. 4, No. 1, March 2004.