Agile Development in Computer Science Education: Practices and Prognosis

Joseph Bergin
Computer Science
Pace University
New York, NY
845-225-4369
berginf@pace.edu

Clifton Kussmaul
(Moderator)
Mathematical Sciences
Muhlenberg College
Allentown, PA 18104-5586
484-664-3352
kussmaul@muhlenberg.edu

Thomas Reichlmayr
Software Engineering
Rochester Institute of Technology
Rochester, NY 14623-5608
585-475-2852
tjrese@rit.edu

James Caristi
Mathematics & Computer Science
Valparaiso University
Valparaiso, IN 46383
219-464-5342
James.Caristi@valpo.edu

Gary Pollice
Computer Science
Worcester Polytechnic Institute
Worcester, MA, 01609-2280
508-831-6793
gpollice@cs.wpi.edu

Categories and Subject Descriptors

D.2.9 [**Software Engineering**]: Management – *productivity, programming teams, software process models*. K.3.2 [**Computers and Education**]: Computer and Information Science Education – *computer science education, curriculum, information systems education*. K.6.3 [**Management of Computing and Information Systems**]: Software Management – *software development, software process*.

General Terms

Design, Documentation, Management, Standardization

Keywords

agility, curriculum, development, methodology, process, software, XP

1. SUMMARY

Agile approaches to software development share a particular set of values [2,4]:

- Individuals and interactions over processes and tools.
- Working software over comprehensive documentation.
- Customer collaboration over contract negotiation.
- Responding to change over following a plan.

SIGCSE'05, February 23–27, 2005, St. Louis, Missouri, USA.
ACM 1-58113-997-7/05/0002.

Many agile methodologies were developed in response to so-called disciplined methodologies that emphasize detailed documentation and formal processes, and that are often associated with ISO compliance or the CMM. However, there is growing recognition that both agile and disciplined approaches have advantages, and that often a combination can be very effective [1].

Many faculty are exploring and experimenting with ways to integrate agile concepts and practices into academic programs in areas such as computer science, software engineering, and information systems. This special session will help us work together in agile ways to better understand the importance and role(s) of agile concepts and practices, successful ways to incorporate them in academic settings, potential pitfalls, and key questions that should be explored further. We want to gather input from a wide range of people in different sub-disciplines and programs.

We will begin with a very brief overview of agile concepts and practices, followed by brief statements from each of the five speakers, to give other participants a sense of the range of possibilities (25 min). Next, we will poll participants to identify a set of topics within agility that they want to discuss further (5 min). Participants will then gather into subgroups for each topic, and each subgroup will identify best practices, interesting ideas, and open questions for that topic (30-35 min). Each subgroup will then give a brief report to the entire group, and we will conclude with a few minutes of general discussion (10-15 min).

2. JOSEPH BERGIN

Many of the techniques of agile methods are beneficial to novices as well as to professionals. In fact the practices of XP are just good practices that can be taught early. It is also true that some of the tools and techniques of XP can be used to enhance the teaching process itself. One such tool is the GeneralFixture of Fitnesse [3], which permits an instructor to present a Java exercise to a student as an executable

specification. This specification can be easily interleaved with natural language requirements. The tool then provides an executable test of the correctness of implementation of an individual requirement. Moreover, the students can easily add additional tests to help them get the program correct. The executable specification is equivalent to a test in JUnit, but it doesn't involve programming so is accessible to students earlier in their education.

GeneralFixture is an optional component of Fitnesse, which is itself a convenient wiki server that also permits asynchronous communication within a course. The tests in Fitnesse are executed directly within a web page by pushing a button on the page and are developed online by filling in a web form. Tables use a simpler syntax than HTML and can even be pasted from Excel spreadsheets.

3. JAMES CARISTI

A number of students resist using some of the agile practices that we try to teach them. In particular, strong students occasionally have difficulty accepting test first and pair programming. Research in pair programming has indicated a few promising strategies. And the experience of several instructors with test first has uncovered a few ideas that seem to work and some that definitely do not.

4. CLIFTON KUSSMAUL

I have used agile practices successfully in consulting projects, and I am experimenting with agile practices in a variety of educational contexts. I encourage, but don't yet require, pair programming in CS I. I find that Iterative development of software projects and written proposals is very effective in both non-majors courses and capstone courses. The emphasis on individuals and interactions, and the lack of detailed formal processes, give students an incentive to reflect on and adjust the process, rather than blindly following it. Multiple iterations and the emphasis on responding to change give students more opportunities to see the interactions between activities, and encourage students to react to and recover from errors early in the project. For me, these advantages are just as important as helping students become more effective developers. I am also trying to show students that many of these ideas are more broadly applicable; for example, agile concepts and techniques can be used to improve student writing.

5. GARY POLLICE

Should we develop agility or analytical skills? Are they distinct?

The Agile Software Development movement has matured and become an effective approach to software development for many industrial environments. However, it is but one approach of several that might be selected. The key skill is to understand what the benefits and disadvantages of each approach are and how and when to apply them.

Agility has a lot to offer when teaching CS courses, especially those that are designed to give students basic programming and problem-solving skills. Pair-programming and test-first practices are useful skills to know. As we progress to software engineering courses, we are faced with the need to broaden the student's skills to more than just agile methods. The question I am continually faced with is whether to just present an agile method like XP, for the students to apply to their, necessarily small, term project, or to give them the knowledge that will allow them to select the right set of practices for their context. I choose the latter and find that this almost always works better in the long run.

As educators we need to find the right ways to introduce students to agile practices while preparing them for all kinds of projects they might encounter in their professional careers. Just teaching agile methods is not sufficient for software engineering courses.

6. THOMAS REICHLMAYR

Can an agile software development process be successfully integrated into an undergraduate software engineering curriculum? The introductory software engineering class at RIT has successfully been using an agile process for the past two years in its term long project. As this course represents the first opportunity for students to participate in a team project, the collaborative nature of agile processes has facilitated the challenges of students transitioning from individually focused developers to contributing team players.

The agile process attributes that have been most beneficial to student teams has been the development of project requirements and a raised quality awareness using test driven design.

Time to delivery is always a major challenge of any software project, but even more so in an academic environment. Student teams have a short period in which to deliver their final release of the project. User stories provide a process in which requirements can be elicited and incremental release strategies developed to evolve the product over the term. Exposure to user story prioritization, scheduling and release planning are perhaps the most valuable skills our students leave the course with.

Testing is identified as a skill that students are least prepared for upon graduation and entry into the industrial workplace. Academic project scheduling pressures often short change the testing process by the students' perception that testing is an end of the waterfall activity that is often consumed by the need for additional implementation (coding) effort. Test driven design in concert with a well conceived project release plan puts the focus on continuous working software through out the project life cycle.

7. REFERENCES

[1] Boehm, B. and Turner, R. Balancing Agility and Discipline: A Guide for the Perplexed. Addison Wesley, 2003

[2] Cockburn, A. Agile Software Development. Addison Wesley (2003)

[3] Fitnesse. http://www.fitnesse.org

Highsmith, J. Agile Software Development Ecosystems. Addison Wesley (2002)