Proceedings of Student-Faculty Research Day, CSIS, Pace University, May 8th, 2009

Evaluating Agile Principles in Active and Cooperative Learning

John C. Stewart, Carolyn Sher DeCusatis, Kevin Kidder, Joseph R. Massi, and Kirk M. Anne

jstewart@lec.edu, dcusatis@uptonline.net, kevinkidder@gmail.com, josesph.massi@pace.edu, kma@geneseo.edu

Seidenberg School of CSIS, Pace University, White Plains, NY 10606, USA

Abstract

This paper presents a review of the literature on the use of agile teaching methodologies, and shows how agile methods are applied to pedagogy. We then define active and cooperative learning to demonstrate that agile teaching methodologies fall in this category. We then attempt to map agile principles as outlined in the Agile Manifesto to specific pedagogical methods and activities. We propose that this congruence can act as a framework for the application of agile practices to education, which can improve teaching effectiveness and facilitate learning and retention among students. We attempt to measure this result directly by surveying DPS students, who are participating in an agile dissertation process.

1. Introduction: Agile Methods in the Classroom

The Agile Alliance, a group of representatives from Extreme Programming, SCRUM, DSDM, Adaptive Software Development, Crystal, Feature-Driven Development, Pragmatic Programming, and others sympathetic to the need for an alternative to documentation driven, heavy-weight software development processes, created the Agile Manifesto [1] to outline agile methods in software development. Agile methods of development have moved into the academic arena with a number of studies on the teaching of agile methods of software development in college level engineering and computer science courses [2, 3, 4, 5].

However, it is not just the software community that has grave concerns that their methodology is too heavy-weight; the academic community has been working for many years on developing techniques to increase student participation, rather than using planned curricula based on instructor lectures. In the education literature there is considerable documentation on active and cooperative learning methodologies and their effectiveness in facilitating student learning outcomes [6].

As the teaching of agile methods of software development moved into the classroom, some instructors began to experiment with using the agile methods to deliver the course material to the students. Reed [7] in a Systems Analysis and Design course minimized the use of lecture, started each class session with a standup and ended each class with a retrospective, allowing students the opportunity to offer immediate feedback. She encouraged interaction among students and the sharing of learning experiences, and used questions, storytelling and writing stories. Reported results included highly motivated students and very favorable feedback from students. Her conclusions support the competence-based approach to using agile principles and practices in the college classroom.

Chun [8] used an agile teaching/learning methodology in computer science courses with the goal of knowledge sharing and nurturing self-learning. He compared the similarities of the teaching process to software development, noting that each involved a number of stakeholders with differing and competing objectives, a tight schedule, fixed deadlines, limited resources, and considerable unexpected changes along the way.

The iterative cycle included monitoring and adjustment to the teaching cycle and practice, independent study, and sharing to the learning cycle. He used weekly quizzes as a measure of student progress. He used open ended assignments to allow for creativity in student practice. Knowledge sharing was a large component of the methodology. He set up time for students to share what they learned and what they were working on and teach others about some component of the project – a learn by sharing approach. The goal was to use independent study as a catalyst to prepare students for life-long learning.

Alfonso and Botia [4], in a Java based software development course, placed the instructor as a manager (similar to the Scrum Master) for each team and provided the students with feedback on each iteration. The instructor provides feedback about errors and students correct them on a subsequent iteration. Their approach is group-work centered and software project based [4]. The researchers reported a “more satisfactory experience” for both the teachers and students.

Layman, et al. [9], used Felder-Silverman learning styles and hands-on student team collaboration in a software engineering course to address the learning needs of students. The researchers believe that the use of agile methods and an awareness of learning needs can help create a successful learning environment.

Razmov and Anderson [10] described an agile teaching methodology applied to a project-based software engineering course. They determined that feedback was important, not only for learning, but also for the ability to adapt. The researchers saw frequent feedback as creating opportunities for both teachers and students. Their view of the agile teaching philosophy is based on the instructor actively seeking student feedback and reacting quickly to bring more relevance to topics, all while seeking the goal of effective student learning. They used a number of feedback mechanisms

including post-milestone project meetings, in-class retrospectives, and student reflective writing. Based on student ratings, they pronounced their methods successful.

In the education literature, the majority of the research on active and cooperative learning exists in journals devoted to general education. There is limited research on the application of agile methods to active and cooperative learning activities in introductory computer science courses as we have shown above. We did not locate specific studies on the use of agile methods in educational or teaching environments or the application of agile principles to pedagogy with respect to active and cooperative learning methods in non-computer science college level courses. However, we observe that some educators are in fact using agile methods of teaching under the framework of the educator's view of active and cooperative learning.

In this paper we will explore the use of agile principles in the development of pedagogical activities to facilitate learning by mapping the ideals and principles of the Agile Manifesto to pedagogical methodologies. We will use this framework to suggest practices that can make teaching more agile and, therefore, more student-centric and effective. We will also survey DPS students, who are participating in an agile dissertation process, and use these results to measure the effectiveness of an agile learning environment.

At first glance, the similarities between the software development methodologies and the educational methodologies are easily seen. Both teaching and software development require detailed planning and scheduling. Each requires management and constant assessment and feedback from all involved. Making sure a course is delivered correctly and on time presents similar difficulties to those encountered in software development projects.

2. Active and Cooperative Learning

Analysis of the academic literature suggests that in order for students to learn, they must take an active approach to their education [11]. This means that they must do more than simply listen. They must discuss, engage in problem solving, read, and write [16]. In order to be actively involved in the learning process, it is suggested that students must participate in higher level thinking through analysis, evaluation, and synthesis [11]. Given this structure, active learning can be defined as instructional activities in which the students "actively" participate (doing things) while thinking about what they are doing. The software engineering courses which used agile methods described in the prior section certainly fall into this category. Bonwell and Eison conclude that several studies point to active learning techniques as having a large impact on student learning. According to Bonwell and Eison, student achievement in active learning based course activities is comparable to lectures in terms of mastery of content. However it is superior to the lecture format in terms of promoting student thinking skills [11].

Sharan concludes that cooperative learning is more effective than individual learning:

"A number of researchers have focused on the quality of reasoning strategy used within competitive, individualistic, and cooperative learning situations. They

conducted a series of studies comparing student performance with competitive, individualistic, and cooperative learning situations on tasks that could be solved using either higher- or lower-level reasoning strategies. They found a more frequent discovery and use with the cooperative condition of such higher-level reasoning strategies as category search and retrieval, intersectional classification, formulating equations, sequencing, metaphoric reasoning, and conservation strategies." [12]

Cooperative learning can be defined as a subset of active learning in which students actively participate in tasks as a group of three or more [13]. Students can also be paired for certain classroom learning activities in a similar way that programmers are paired for problem solving in software development. Sharan suggests that teams succeed more than individuals [12].

Sharan stresses four key points to cooperative learning: interdependence (both goal and reward oriented), promotive (face to face) interaction, personal responsibility/individual accountability and group processing [12, 14].

While agile methods are clearly active and cooperative, a case may be made that many active and cooperative learning techniques are also agile.

Kagan and Kagan explored what is required in order for teams to work in a structured environment of cooperative learning. One of the key team elements is how the teacher handles management issues [14]. The teacher will work with each team as if they were a part of the team, similar to a customer being available on-site for a software development team.

As with the agile model, feedback is an important component of active learning methodologies. In addition to the "learn-by-doing" aspect of active learning, activities that incorporate student participation also allow for feedback to the instructor to determine understanding of the material.

3. Active and Cooperative Learning and Student Success

When students are actively engaged in the college experience, learning and retention are more likely [15]. In a landmark study, researchers determined that seven engagement indicators directly influenced the level and capability of student learning and their overall experience at a college. Five of these factors were policies of the faculty [16]:

- Encouraging cooperation among students
- Encouraging active learning
- Communicating high expectations
- Encouraging contact between students and faculty
- Using active learning techniques

Much of the research has shown a strong association between heightened student learning and formal and informal faculty-student interaction and contact [17]. This interaction then, essentially controlled by faculty, is often the best predictors of student persistence. Consequently, the higher the degree of interaction, the greater the likelihood of student engagement and success.

4. Mapping the Agile Manifesto to the Classroom Environment

The Agile Manifesto identifies the following values:

Individuals and interactions over processes and tools
Working software over comprehensive documentation

Customer collaboration over contract negotiation
Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more. [1]

Modern teaching methods value students and instructors and their interactions over a specific approach or method of teaching and learning, working knowledge over rote memorization, communication rather than negotiation, and being receptive and responsive to changes rather than adhering to a specific schedule.

In terms of teaching effectiveness we propose the following values for Agile Pedagogy, applying the Agile Manifesto as a template:

Students over traditional processes and tools. Student-centric learning environments should be favored rather than the traditional lecture-driven environment. Students actively participate in the learning process through activities and group-based components that reinforce concepts and allow for exploration.

Working projects over comprehensive documentation. In project based courses, projects are due near the end of the semester and are often presented during the last class meeting. Students often wait until the 11th hour to begin the project. Working in an iterative environment with a deliverable component of the project required each iteration, leads to greater immersion in the project, more learning, and a better quality final deliverable.

Student and instructor collaboration over rigid course syllabi. In a student (customer) – centric environment the focus is on what the student is doing and what pedagogical methods are working to facilitate learning. In many courses the syllabus is outlined as the contract between the student and the instructor. A more flexible relationship with greater access to the instructor (who is the customer in some cases) can lead to more collaborative relationship. For example, in a spreadsheet model development course, access to the instructor via e-mail to offer suggestions, and to ask probing questions about the student's model is an iterative and agile approach to active learning.

Responding to feedback rather than following a plan. Often instructors outline the course to the students with specific topics to cover on specific days with fixed due dates. In addition instructors have a standard method of delivering the material and are often unwilling or incapable of adapting the delivery methods to the class dynamic. Every class is different and each may require different delivery methods and timing in order to facilitate learning outcomes. Agility is the ability to adapt to different learning styles and change the delivery methods if the current methods are not producing the desired results. Interaction and feedback are important. Students need to know how they are doing and what is expected and instructors need to know how students are responding to the current teaching approach.

5. Mapping Agile Principles to the Classroom Environment

Principles of the Agile Manifesto	Corollary to the Pedagogical Environment
Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.	Our highest priority is to prepare the student to contribute to an organization through continuous delivery of course components that reflect competence.
Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.	The instructor and students welcome and adapt to changes even late in the semester. Agile pedagogical methods use problems and change as an opportunity to facilitate learning and better develop marketable skills in the students.
Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.	Requiring working deliverables from the students over short time periods allowing for frequent feedback and guided problem solving and experimentation.
Business people and developers must work together daily throughout the project.	There is iterative interaction between the instructor and students (or student groups) during each iteration of course components.
Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.	Trust that most students are motivated. Give them the environment and support necessary that for them to be successful.
The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.	To the extent possible, allow for direct face to face interaction with students or student groups.
Working software is the primary measure of progress.	Working deliverables (i.e. models, software, project deliverables, presentations, etc.) are the primary measure of student progress (not necessarily midterm & final exams that require rote learning and memorization).
Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.	The cooperative learning environment where students actively seek guidance and tools to solve problems is the basis for teaching the skills needed for life-long learning.
Continuous attention to technical excellence and good design enhances agility.	Continuous attention to technical excellence and good design enhances learning.
Simplicity--the art of maximizing the amount of work not done--is essential.	While in education there is some value in exploring subjects in depth just because there is student interest, understanding the problem and solving it simply and clearly is essential.
The best architectures, requirements, and designs emerge from self-organizing teams.	Student groups and teams should self organize, but all should participate equally in the effort.
At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.	At regular intervals, the students and instructor reflect and offer feedback on how to be more effective. All stakeholders then adjust accordingly with the goal of being more effective.

6. Connection to Agile

The correspondence between elements seen in the agile software development environment and the academic environment are striking. Judy Clarke outlines the "Jigsaw Method" to cooperative learning [18]. The Jigsaw method shares many practices with XP and Scrum in the agile concepts in that the Jigsaw Method depends on groups. The groups (or agile teams) depend on one another to achieve tasks. The members of effective groups bring together diverse strengths, interest, expertise, experience, knowledge, perspective and personalities to attain goals that surpass those that can be achieved by individual members.

In many ways the Jigsaw approach is similar to the pairing concept in that with the Jigsaw method, there is no "getting stuck". The inter-reliance on each other's capabilities and insight brings the group together**.** [19]

According to Sharan, in order to incorporate cooperative learning methods into instruction, teachers must look at their role as more than simply transmitters of information and view themselves as guides and facilitators [6]. The purpose of the group is inquiry or problem solving. The interaction and motivation of the group members is an important component of the success of the team. It is interesting to note some of Sharan's views of the role of the teacher and how they apply to agile principles:

- Offers help to groups that need it.
- If members of a group are unhappy with their original plan, the teacher discusses alternatives rather than insisting on sticking with what may be an uninteresting or unworkable plan.
- Leading a summarizing discussion.
- Making sure all group members participate.

7. Measuring the Results of the Agile Dissertation Process

We are surveying the Pace University's DPS student population, asking them ten questions about the agility of their experience and how their graduate experience was affected by the agile process. The results of this survey will be presented on Research Day.

8. Conclusion

As we have outlined in our research, many parallels exist between agile software development and active and cooperative learning. Both methodologies value students and teachers and their interaction instead of specific approach to delivering the material or (from the perspective of the student) learning. Each desires the development of capability or competence or working knowledge over rote learning. Both depend on effective communication in place of the negotiation based relationship. Teaching and software development are more effective when response to changes are viewed as part of the process rather than the unwavering adherence to a specific schedule.

Agile teaching attempts to encourage and address student questions. The curriculum is project based (both large and small), and the emphasis is not on the lecture, but on student learning experiences while working

collaboratively on projects and activities. Agile teaching is about working closely with students and a willingness to be flexible to meet their needs and responding to those needs. There is direct involvement of the students in the learning process. Delivering the course material agilely means to be goal driven as opposed to plan-driven as is the case with more conventional course design.

Our review of the literature agile and active/cooperative learning literature showed parallels between the utilization of agile methods in teaching (pedagogical methods) and active/cooperative learning techniques commonly found in the education literature. We propose that each of these components complements each other, and that using agile methods of delivering the material combined with active learning activities, leads to improved learning outcomes and higher levels of student engagement and satisfaction.

References

1. Agile Manifesto http://agilemanifesto.org/ .
2. Hislop, et. al. "Integrating Agile Practices into Software Engineering Courses", **Computer Science Education**, Volume 12, Issue 3 (September 2002) , pages 169 – 185.
3. Boehm, Barry; Port, Dan; Brown, A. Winsor. "Balancing plan-driven and ***agile*** methods in software engineering project courses", **Computer Science Education**, Volume. 12 Issue 3 (September 2002) p. 187-95.
4. Alfonso and Botia, An Iterative and Agile Process Model for Teaching Software Engineering, **Proceedings of the 18th Conference on Software Engineering Education & Training,**(CSEET'05), 2005.
5. Bergin, Kussmaul, Reichlmayr, Caristi, and Pollice, "Agile development in computer science education: practices and prognosis",**SIGSCE** '05, ACM.
6. Sharan and Sharan, "Group Investigation in the Cooperative Learning Classroom," **Handbook of Cooperative Learning Methods**, Sharan, Ed., Greenwood Press, Wesport, CT, 1994.
7. Reed, An Agile Classroom Experience, Agile 2008 Conference, IEEE Computer Society, 2008.
8. Chun, The Agile Teaching/Learning Methodology and its e-Learning Platform, **Lecture Notes in Computer Science-Advances in Web-Based Learning**, Volume 3243/2004, Springer-Verlag Heidelberg, pp.11-18.
9. Layman, Cornwell, and Williams, Personality Types, Learning Styles, and an Agile Approach to Software Engineering Education, **SIGCSE** '06, ACM, March 1-5 2006, Houston, TX.
10. Razmov and Anderson**,** "Experiences with Agile Teaching in Project-Based Courses" **ASEE**, 2006.
11. Bonwell and Eison, Active Learning: Creating Excitement in the Classroom, **ERIC Digest**, IRIC Clearinghouse on Higher Education, George Washington University, Washington, DC, 1991.
12. Sharan, Shlomo, **Cooperative Learning: Theory and Research**, Praeger Publishers 1990.
13. Paulson and Faust, "Active Learning for the College Classroom," **Journal on Excellence in College Teaching**, 1998, 9(2), 3-24.
14. Kagan and Kagan, The Structural Approach : Six Keys to Cooperative Learning, **Handbook of Cooperative Learning**, Shlomo Sharan, Ed. , Greenwood Press, 1994.
15. Tinto, **Rethinking the Causes and Cures of Student Attrition**, 2nd Ed., University of Chicago Press, Chicago, 1993.
16. Chickering and Gamson, Seven Principles for Good Practice in Undergraduate Education, **AAHE Bulletin** 39:3-7. ED282 491, Mar 1987.
17. Umbach and Wawrzynski, Faculty Do Matter: The Role of College Faculty in Student Learning and Engagement, Research in Higher Education, 46(2), 153–184, 2005.
18. Clarke, Pieces of the Puzzle: The Jigsaw Method, **Handbook of Cooperative Learning**, Shlomo Sharan, Ed. , Greenwood Press, 1994.
19. Simon and Hanks, "First Year Students impressions of pair programming in CS1", ACM **Journal on Educational Resources in Computing** (JERIC) Volume 7 no. 4 2008.