Resources for Agile Software Development in the Software Engineering Course

Laurie Williams, Sarah E. Smith, Michael Rappa
North Carolina State University
{lawilli3, sesmith5, mrappa} @ncsu.edu

Abstract

Agile software development is an emerging topic in software engineering. Industrial use of agile techniques is growing faster than our educational resources are being updated. Using an open courseware platform, OpenSeminar, we provide course resources for software engineering students about agile software development methodologies. These resources include readings, sites, lab exercises, lectures, examples, testing practices, and testing frameworks.

1. Problem Statement

Agile software development is an emerging topic in software engineering. The need to educate students in agile practices and processes is specifically addressed in the ACM/IEEE Computing Curricula 2001 (CC2001) Software Engineering Education Knowledge (SEEK) document [1]. Additionally, industrial use of agile techniques is growing faster than our educational resources are being updated. These trends present educators with two challenges. First, educators must learn new techniques. Secondly, educators must provide resources to students. However, educational materials provided in textbooks lag behind practical advances in the use of agile techniques.

2. Open Seminar

We have created resources to support education in agile practices. These resources are appropriate for a student who has completed a CS1 and CS2 course. We provide our resources via OpenSeminar (http://openseminar.org/), a web-based open courseware platform architected by the third author. OpenSeminar is designed to enable professors to collaborate on material for similar courses. Initially, one professor can create and populate a course with resources. Subsequently, professors at other universities can adapt the resources of the initial course instance to meet their own needs. The adaptation includes choosing which of the posted resources would be valuable for their own classes and adding additional resources. When a professor adds a resource to OpenSeminar, all collaborating professors have visibility to this resource and can choose to suggest this resource to their students. The end result of an OpenSeminar in a given subject is an expert-maintained repository or database of links to online information that is openly available on the Internet.

The first general course created in OpenSeminar is the OpenSeminar in Software Engineering (OSSE, http://openseminar.org/se/), based on the software engineering course taught by the lead author at North Carolina State University (NCSU). The OSSE contains several modules with courseware related to agile development practices in addition to other more traditional techniques. In total, there are 22 course modules in OSSE, six of which relate to agile software development. Each module provides readings, sites, lab exercises, lectures, and examples. Some specifics of these six modules are provided in the next section.

Proceedings of the 18th Conference on Software Engineering Education & Training (CSEET'05)
1093-0175/05 $ 20.00 **IEEE**

3. Agile Methodology Courseware

There are six OSSE modules which relate to agile software development: software process, pair programming, testing, requirements, project management, and the comprehensive Monopoly example. The intent of these resources is to increase the breath of possible approaches to a project to include agile alternatives. Some details of these six modules are now provided. The first five of these modules relate to knowledge units of CC2001 SEEK; these knowledge units will references in this discussion.

- **Software process.** General resources are provided for both agile and plan-driven software development methodologies. Specifically a 20-page survey of four agile methods (written by the first author) is provided (Extreme Programming (XP), Crystal, Scrum, Feature-Driven Development) in addition to other readings, a lecture, and links to sites. The SEEK specifically notes that agile software development should be included in lessons about lifecycle models (PRO.imp.2).
- **Pair programming**. Several interactive lab exercises are provided. Readings and other sites related to pair programming are provided. The SEEK does not specifically mention pair programming. However, pair programming is increasingly being considered as a validation and verification (V&V) practice, akin to desk checking (VAV.rev.1), walkthroughs (VAV.rev.2), and inspections (VAV.rev.3). Additionally, pair programming provides students experience with group dynamics (PRF.psy) and communication skills (PRF.com).
- **Testing.** The XP automated testing practices of test-driven development and acceptance testing are presented in detail in addition to more fundamental white-box and black-box testing techniques. Test-driven development is specifically mentioned in the Computing Essentials section of the CC2001 SEEK, CMP.ct.16. An extensive reading on the automated testing techniques is provided in addition to tutorials, interactive lab exercises, other readings, and links to sites. As will be discussed in the Monopoly example below, a complete working program with automated tests is provided.
- **Requirements**. The requirements module provides resources on three different means of recording requirements: traditional, use case-based, and user story-based. User stories are the means of recording requirements in the XP methodology. In addition to readings on these three methods, the Monopoly example requirements were done each of the three ways, enabling comparison. The readings explain more about when each of the methods would be most appropriate. Also provided are other readings, additional links, lab exercises, and tutorials. The CC2001 SEEK document teaching agile-style requirements process fundamental (MAA.rfd.2) and requirements management (MAA.rfd.6).
- **Project management**. An interactive lab exercise is provided which features the Extreme Hour that introduces students to the practice of writing user stories and managing their project via XP release planning. The CC2001 SEEK discusses the need to apply project planning (MGT.pp) and project control (MGT.ctl) to projects that use agile methodologies.

- **Monopoly Example**. A Java implementation of a Monopoly game is provided in addition to a myriad of supporting artifacts: working code, automated JUnit[1] unit test cases, automated FIT[2] acceptance test cases, black box/functional test plan, class diagram, sequence diagram, state diagram, and requirements specification (three complete version for each means of documenting requirements)

Our collection of agile methodology courseware covers several tools and resources that are not yet covered in general software engineering textbooks. Other educators could use our courseware to supplement current resources and/or textbooks in order to keep students up to date about emerging ideas and technology in the computer industry. By collecting together the best resources about agile methodologies, we have provided professors with a single place to go to find resources to supplement their current courseware.

References

[1] ACM/IEEE-CS Joint Task Force on Computing Curricula, "Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering," August 23, 2004.

[1] http://www.junit.org/
[2] http://fit.c2.com/