Open Course Resources as Part of the OpenSeminar in Software Engineering

Michael Rappa, Sarah E. Smith, Alex Yacoub
North Carolina State University
{mrappa, sarah_smith, aryacoub}@ncsu.edu

Abstract

Software engineering is a rapidly changing field and new ideas, like agile software development, are emerging. As such, these ideas must be incorporated into software engineering courses so that students will be up-to-date on current innovations. To help address this problem, we have created the OpenSeminar in Software Engineering. For the OpenSeminar in Software Engineering, we have created course materials that cover various, essential Software Engineering Education Knowledge areas. This paper presents the Monopoly example; comprehensive software engineering courseware and part of the OpenSeminar in Software Engineering.

1. Introduction

Software engineering is a rapidly changing field and new ideas, like agile software development [1], are emerging. Online course resources are a vital medium for sharing cutting-edge knowledge with students [2]. However, searching for these resources and keeping them updated is time consuming. Software engineering professors could benefit from collaboration on openly available resources for use in software engineering courses.

To address this challenge, we have created OpenSeminar (http://openseminar.org/), an open-source web-based application that allows experts to collaborate in the gathering and organization of links to course materials. In this paper, we present the OpenSeminar in Software Engineering (OS/SE) (http://openseminar.org/se). The repository of course resources for OS/SE was populated through an extensive search of the Internet for openly-available, relevant course materials. Other course materials were created in part by the second author. One of the major resources, the Monopoly example, provides a comprehensive illustration of requirements, design, code, and test artifacts across various agile and plan-driven software engineering development methodologies.

2. OpenSeminar in Software Engineering

OpenSeminar is an open-source, web-based application that allows experts to collaborate in the gathering and organization of links to openly available course materials. OpenSeminar adopts the model of a scholarly journal, allowing a subset of recognized experts in a field to voluntarily join together, in the form of an editorial board, to oversee the creation of a repository of educational materials. A professor, who becomes a member, can then use OpenSeminar to select a subset of those resources and deliver the results dynamically to his or her students via a customized course website.

An installation of OpenSeminar in a given subject is in effect an expert-maintained repository or database of links to organized online information that is openly available on the Internet. Materials chosen for OpenSeminar should represent the breadth and depth of a particular subject. As innovations occur in a subject, courseware materials will be updated, and OpenSeminar users will always benefit from the most current resources.

The repository of course resources for OS/SE was populated through an extensive search of the Internet for openly-available, relevant materials and by development of course materials by the second author, particularly in the area of agile software development [1]. The capstone of the developed course materials is the Monopoly example. The Monopoly example is an illustration that incorporates agile development practices and traditional software development practices in one comprehensive example and is part of the larger repository of software engineering educational materials in the OS/SE.

3. Monopoly example

The Monopoly example is a comprehensive look at several software engineering knowledge areas that is incorporated into a basic Java implementation of a Monopoly game. The second author and another graduate developed the Monopoly program using test-driven development in the Eclipse (http://www.eclipse.org/) open-source integrated development environment. Traditional and agile software engineering development methodologies are illustrated via appropriate documentation.

This example incorporates several documents from the requirements phase of development. These documents include a traditional requirements specification, use case requirements, and user stories. By presenting requirements documents based on a common board game, students can see how requirements are created, and compare the different types of requirements.

At the design level, the Monopoly example provides a class diagram generated from the code, one sequence diagram, and a state diagram. All of the diagrams were created using Eclipse UML (http://www.eclipseuml.com/), an Eclipse modeling plug-in. They can be viewed in Eclipse via the Monopoly modeling project or online (http://open.ncsu.edu/se/monopoly/).

The source code and a full suite of test cases are provided with the Monopoly example. There are JUnit (http://www.junit.org) automated unit test cases, and FIT (http://fit.c2.org) automated acceptance test cases included with the Monopoly example code. A test plan document is also provided.

The Monopoly example is provided on OpenSeminar as part of its own module and the resources are also provided as components of the Requirements, Design, and Testing modules and sub-modules. Professors may use the Monopoly example by downloading the resources to use for their own class or by creating and customizing a course inside OpenSeminar to contain the Monopoly resources.

3.1. Monopoly example in relation to SEEK 2004

The Monopoly example provides a broad illustration of many software engineering concepts, and therefore applies to several Software Engineering Education Knowledge (SEEK 2004) [3] areas as outlined below:

- Computing Essentials (CMP): the Monopoly example includes the source code and test code and therefore may be used as an example of several fundamental computing concepts, including error handling, user interface design, testing, etc.
- Software Modeling and Analysis (MAA): the Monopoly example includes several different design diagrams (class, sequence, state, and use case diagrams) that describe different parts of the system. Different types of requirements resources (traditional, user stories, and use cases) are used to represent how different requirements methods can represent the same product.

- Software Design (DES): the design modules used in the Monopoly example represent object-oriented design.
- Software Verification and Validation (VAV): the Monopoly example includes a full suite of JUnit and FIT tests. This testing effort is documented in a test plan.

The Monopoly example, or parts of the example may be adapted to many different courses as outlined in SEEK 2004 document [3], particularly any course with the SE designation. The example may be broken down into smaller parts and used in courses that cover a range of topics, for example in the core software engineering sequences. By providing one example across several different courses, students will be able to understand how each individual course fits in with the larger picture of software development.

3.2. Use of Monopoly example

The primary objective for using the Monopoly example is to provide students with a complete program, related documentation, and tests to be used in the classroom to teach concepts or as a reference. As such, there are no interactive learning activities associated with the Monopoly example. The Monopoly example may be used as a teaching resource or reference materials, and therefore students do not need to be familiar with the covered materials before using the Monopoly example.

While the Monopoly example provides all of the required resources, the example may be adapted for more interactive use. For instance, a professor may provide the requirements to students for the creation of different design diagrams or implementation of the program in the language of the professor's choice. The code may be provided for students to test or analyze.

4. Conclusion

We have presented the Monopoly example; a comprehensive illustration of a number of software development activities that cover several software engineering knowledge areas. This example is available for use online as part of the OS/SE repository of software engineering course resources.

5. Acknowledgements

The authors would like to thank the Open Courseware Lab (http://open.ncsu.edu) for development of the OpenSeminar platform and Chih-wei Ho for his part in developing the Monopoly example.

6. References

[1] L. Williams, S. E. Smith, and M. Rappa, "Resources for Agile Software Development in the Software Engineering Course," in *Conference on Software Engineering Education & Training*. Ottawa, Canada: IEEE Computer Society, 2005.

[2] R. Baraniuk, G. Henry, and B. Hendricks, "Peer to Peer Collaboration with Connexions," in *EDUCAUSE 2004 Annual Conference*. Denver, Colorado, 2004.

[3] The Joint Task Force on Computing Curricula, "Software Engineering 2004: Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering," vol. 2006: IEEE Computer Society and ACM, 2004.