A Life-Cycle Engineering Case Study

Thomas B. HILBURN, Massood TOWHIDNEJAD, Salamah SALAMAH
Department of Electrical, Computer, Software, and Systems Engineering
Embry-Riddle Aeronautical University
Daytona Beach, FL, USA

ABSTRACT

For engineering degree programs, one of the primary objectives is to teach engineering practice in a course-by-course academic structure. Although engineering programs typically have a year-long senior level design project to provide engineering development experience, it is still a challenge to provide students with the full life-cycle experience and expose them to all of the various dimensions of engineering practice. This paper advocates the development and use of a full life-cycle case study to address such a challenge. A software engineering case study, developed by the authors, and used in various courses and workshops, illustrates the nature, structure and value of this approach. The case study learning objectives emphasize development of engineering practice and acquiring team, problem solving, and analysis skills. The paper also describes how this life-cycle case study can be used throughout a software engineering curriculum.

Keywords: case study teaching, engineering education, software engineering

1. INTRODUCTION

Although the use of case study teaching has proven its worth and is a widely used method of teaching in fields such as business, law, and medicine, it is yet to be accepted to the same extent in engineering education, except in the area of engineering ethics. One of the reasons for this is the lack of sufficient material for this purpose. Many engineering textbooks provide example cases to illustrate concepts and techniques, and there are various websites (e.g., http://www.afit.edu/cse/cases.cfm, http://sciencecases.lib.buffalo.edu/cs/) offering case studies of engineering and scientific work. However, they often lack the following:

- Realistic artifacts (often space or intellectual property concerns do not allow one to provide a complete engineering artifact such as a design document or a project plan)
- Completeness (most are focused on some part of engineering practice, or on a single course)
- Ability to decouple from the intended use and apply in ways not intended by the author
- Techniques for integration into course activities,
- A scenario format that motivate students' engagement in problem identification/solution.
- Guidance to the instructor on how to use the case study material

Engineering is concerned with the application of science and mathematics to design and develop products and services that serve the needs of humankind. By its nature, it is a "problem-solving" discipline – that is, its practitioners (engineers) develop solutions to problems posed by the stakeholders in an engineering project. The typical undergraduate engineering curriculum is a four to five year program where engineering students first study foundation material in mathematics and in the natural and engineering sciences. This is followed by courses that involve the study and application of engineering and design principles and apply them in a capstone engineering design project. The engineering and science courses are supplemented with study in communication, engineering economics, project management, and other general education topics.

In a recent study [1], an engineering competencies survey of over 4000 graduates, in eleven different engineering disciplines, solicited opinion about the relative value of ABET (Accreditation Board for Engineering and Technology) prescribed student outcome areas. The four highest rated competencies were the following: "ability to function on a team", "engineering problem-solving skills", "ability to analyze and interpret data", and "written and oral communication skills". We will discuss how case studies can be effective in addressing these competencies

2. ENGINEERING CASE STUDIES

ABET [2] requires that all engineering programs involve their students in "a major design experience based on the knowledge and skills acquired in earlier course work and incorporating appropriate engineering standards and multiple realistic constraints". Students are typically grouped into teams to work on a semester or year-long senior engineering development project. Unfortunately, this is too often isolated from the rest of the curriculum and does not form a real-world basis for the entire curriculum. Although, students may be exposed to elements of engineering practice in their foundation engineering courses, they often arrive in their senior project course with disjointed view of engineering practice and have an insufficient appreciation and understanding of the complexity and multifaceted nature of real-world development environments. In addition, students often lack sufficient team, analysis, and communication skills. Therefore, it is imperative that engineering curricula introduce professional and real-world education throughout a curriculum. Many programs now introduce small team-based engineering projects in the early part of the curriculum. Case studies can be used to support and extend such experiences.

An excellent way to address the challenges of introducing real-world exposure through a curriculum is the use of a robust life-cycle engineering case study – that is, a case that engages the student in the engineering of a complex system throughout its "life": system definition, preliminary and detailed design, implementation, verification and validation, and operation and maintenance. Of course the system chosen would vary with the engineering field: an airplane for aeronautical engineering, a suspension bridge for civil engineering, a power grid system for electrical engineering, or a smart house software system for software engineering. Friedman and Sage [3] present a technique for analyzing and assessing the life-cycle features of a system engineering case. We discuss an example of the life-cycle engineering case study approach in the next section.

The literature describes a variety of ways that case studies can be used to support learning and research, sometimes referred to as the "case method" [3, 4, 5]. Case studies should be viewed as active learning tools; they are meant to encourage participation, debate and understanding. Although they can be used in a didactic, teacher-centered pedagogy, they are most effectively used in an active learning, student-centered system where the teacher acts as a facilitator. The case method can be mixed and matched with other pedagogies such as lectures, guided discussions, and project work. Case studies can be used to supply the background needed for specific problems and design projects; serve as subjects for class discussions; or they can be used to motivate further study, and to identify and formulate research problems.

3. A SOFTWARE ENGINEERING CASE STUDY

For several years the authors have been involved in a case study project that focuses on the development of a DigitalHome (DH) system [6]. The DigitalHome Project, when completed, will cover the complete life-cycle development of a software product (project management, requirements analysis and specification, design, implementation, testing and maintenance).

The DH Project is based on a scenario about a real-world, but fictitious, company, HomeOwner, which is the largest national retail chain serving the needs of home owners. Homeowner, based on market and technology research, decides to invest in the development of a "smart" house, the DigitalHome system, which will have the following features:

- The DH system shall allow any web-ready computer, cell phone or similar device to control a home's temperature, humidity, lights, security devices, and the state of small appliances.
- The communication center of the DH system shall be a personal home owner web page, through which a user can monitor and control home devices and systems.
- The DigitalHome shall contain a master control device that connects to the home's broadband Internet connection, and uses wireless communication to send and receive communication between the DH system and the home devices and systems.
- The DigitalHome shall be equipped with various environment sensors (temperature sensor, humidity sensor, power sensor, contact sensor, etc.). Using wireless communication, sensor values can be read and saved in the home database.
- The DH system shall include programmable devices (thermostats, humidistats, contact sensors, and small appliance and lighting power switches), which allows a user to easily monitor and control a home's environmental characteristics from any location, using a web ready device.
- The DH system shall include a DH Planner, which provides a user with the capability to direct the system to set various home parameters (temperature, humidity, and on/off appliance and lighting status) for certain scheduled time periods.

The initial phase of the case study project concentrated on building a foundation for full development: research into case study teaching; identifying a case study problem; creating a scenario framework; describing the launch of the software development team; fashioning a software development plan to guide development of the DigitalHome System; establishing a development process; creation of a DigitalHome need statement; analysis, modeling and specification of the DigitalHome requirements; development of a system test plan; development of a software architecture; and specification of the system components.

We have developed a set of DH mini-case studies (we call "case modules") which are designed to engage students in active learning software engineering activities related to development of the DigitalHome System. With this life-cycle engineering case study, we advocate a "Team Learning Format" [4]. Our approach involves the following activities:

- Students are assigned prior reading or other preparation.
- Instructor introduces the case, providing motivation and giving background.
- Instructor divides class into teams and if necessary, assigns roles.
- Teams work on a case module exercise: discussing problems/issues, answering questions and making decisions, and prepare report of their conclusions.
- Teams present their report to class and class discusses results.

Existing DigitalHome scenarios, artifacts and case modules can be viewed at http://www.softwarecasestudy.org/. The case study material includes a complete set of software development artifacts as well as case exercises that can be used to teach different topics (i.e., requirements, design, programming, testing, quality reviews, project management, etc.) throughout a computer science or software engineering curriculum. Each case exercise represents a mini case study and is associated with a specific teaching subject (e.g. project planning, requirements inspection, object oriented design, system test planning, etc.) and a set of learning objectives.

4. A SOFTWARE INSPECTION CASE MODULE

An early Digital Home case study effort was the analysis and specification of the software requirements for the system, which resulted in a DigitalHome Software Requirements Specification (SRS). In addition to the specification of the functional requirements, the SRS includes a description of user characteristics, development constraints, the performance environment, and nonfunctional requirements specifying performance, reliability, and safety and security requirements. The SRS also includes a use case model. As part of the development of the requirements specification, the team developed a case module for inspection of the SRS. Tables 1, 2 and 3 provide a description of various case module elements. The details of some elements are not included for brevity's sake.

Table 1 provides an overview of the contents of a module. The learning objectives for this case module go beyond assessing the quality of the SRS, but are intended to address critical software engineering education goals: appreciating and understanding the problems in specifying requirements; learning to work as part of a team; using and following an inspection process; and analyzing inspection data. These objectives are consistent with the most valued ABET outcomes mentioned earlier. In fact similar types of learning objectives are part of most of the DH case modules.

Notice the list of DH artifacts and the contents of the inspection package. Although the SRS artifact is the chief focus of the inspection, the others (such as the background scenario and need statement) provide the setting and context to create a realistic environment for a professional and effective software inspection exercise. The inspection package provides the sort of forms and tools used in a mature inspection process and help guide the students not only in performing an effective inspection, but in understanding how a best practice works.

Table 1: SRS Inspection Module Outline

Case Module: SRS Inspection
Prerequisite Knowledge: Understanding of basic elements of a Fagan Software Inspection process.
Learning Objectives: Upon completion of this module students will have increased ability to: [1] Work as a member of an Inspection Team [2] Assess the quality of a Software Requirements Specification(SRS) [3] Describe problems in specifying the requirements for a software product. [4] Work more effectively as part of a team. [5] Explain the inspection process. [6] Describe the value of the Fagan inspection process. [7] Describe how inspection data can be used to assess the quality of a software artifact and the effectiveness of an inspection activity.
Keywords: Customer Needs, Software Requirements Specification, Fagan Software Inspection
Case Study Artifacts: [1] DH Customer Need Statement [2] DH High Level Requirements Definition (HLRD) [3] DH Background Scenario [4] DH Team Biographical Sketches [5] DH SRS, Version 1.2 **Inspection Package:** • Inspection Process Description • SRS Inspection Checklist • Defect Log • Inspection Summary Report Form
Case Study Participants: …
Scenario: …
Exercise: …
Appendices: Exercise Booklet
Resource Information: …
Teaching Notes: …

Table 2 includes a scenario that is part of the SRS Inspection Case Module. Scenarios are used throughout the DigitalHome life-cycle to provide context and some realism to the learning activities associated with DigitalHome.

Table 3 provides a set of teaching notes intended to support use of case modules in undergraduate and graduate software development courses, providing guidance and suggestions to a teacher using the case module.

Table 2: SRS Inspection Module Scenario

Case Module: SRS Inspection
Case Study Participants: • The DH Team & Jose Ortiz, Director, DigitalHomeOwner Division of HomeOwner, Inc.
Scenario: In early September of 2010, HomeOwner Inc. (the largest national retail chain serving the needs of home owners) established a new DigitalHomeOwner division that was set up to explore the opportunities for equipping and serving "smart houses" (dwellings that integrate smart technology into every aspect of home living). … The Marketing Division produced two documents: the *DH Customer Need Statement* and the *DH High Level Requirements Definition* (HLRD). In September 2010, a five person team was assembled for the project and in early October 2010 carried out a "project launch". After project planning was completed the team began work on requirements analysis and specification. The first version, 1.0, was completed in early October and versions 1.1 and 1.2 were completed by mid-October. In consultation with Jose Ortiz, the team has decided to carry out a formal Fagan inspection of the SRS, version 1.2. Jose has agreed to act as a customer on the inspection team, Michel Jackson is the author, Disha Chandra will be the moderator and other roles will be assigned in the overview meeting.

Table 3: SRS Inspection Module Teaching Notes

Case Module: SRS Inspection
Teaching Notes: • This case module could be used in different level courses (from an introductory level course in software engineering to an upper level or graduate course in requirements engineering or quality assurance.). • Assuming an adequate student preparation for the exercise, allowing students about three hours each for the exercise should be sufficient: assuming two hours for preparation and inspection, and one hour for the inspection meeting. • Preparation time can be done as an outside class activity to be reported as part of a deliverable before inspection meeting • It would be beneficial to follow the exercise with a twenty to thirty minute discussion concerning the student team results. Some key points to include in the discussion are the following: < Discuss how closely the inspection process was followed: < How well did the team conduct each phase? < How well did students carry out their assigned inspection role (i.e., moderator, author, inspector)? &lt; … • Student team members should be cautioned about a few things: < Leave their ego outside of the meeting room < Their job is to identify defects, not fix them. &lt; …

The paper "Read Before You Write" [7] reports on the use of the SRS Inspection Case Module in three software engineering classes (two sophomore level classes and one senior level class), in which nine inspection teams were formed to carry out the module exercise. The inspections concentrated on the functional requirement statements and used the Inspection Package described in Table 1. The inspection process and materials are based on the work of Fagan [8] and Humphrey [9].

Table 4 shows the data for one team, which could be considered typical. In the comments section of Table 4 we compare the team's results with benchmark goals for inspection data (based on work by Humphrey [9]). Although the defect removal rate and the overall inspection rate seem reasonable, the team identified only 10 major defects, while the DH project team had previously reviewed the SRS and identified over 20 major defects (many purposely seeded in the SRS). One conclusion might be that the inspection team was ineffective; however, we viewed this more as an education exercise, not strictly a quality assurance activity. By engaging students in a "close" reading of the SRS, we helped them to understand its meaning, to determine the degree to which it addressed the customer need statement, to evaluate the correctness, clarity and precision of the requirements statements, and to identify missing features. In addition, the team, problem-solving and analysis skills were enhanced. We should also note that the collection and analysis of team inspection data provides the teacher with excellent information on how the inspection was conducted and the degree to which the case module objectives were reached. It should be noted that inspections and reviews are a common quality assurance techniques in all fields of engineering.

Table 4: SRS Inspection Team Data

Inspection Features	Team Data	Comment
Requirements Size	7 pages	The team only reviewed a portion of the SRS.
Major Defects Identified	10	Correction of a major defect either changes the program source code or would ultimately cause change in the program source code.
Major Defects Missed	2	Total defects of 12 were estimated using the "capture-recapture" method [9]
Total Inspection Time	14.2 hrs	9.2 hrs for preparation time and inspections time; one hr inspection meeting (times 5 people).
Defect Removal Rate	0.7 def/hr	A benchmark goal is 0.5 def/hr [9]
Inspection Rate	0.49 pg/hr	A benchmark goal is < 2 pg/hr [9]

5. CURRICUM-WIDE CASE STUDY TEACHING

Table 5 presents a framework for coverage of various software engineering practices throughout a software engineering curriculum. The courses are those described in a reference curriculum developed by the Association Computing Machinery and IEEE Computer Society [10].

Each of case study areas, listed in Table 5, would have at least one DH software artifact and one or more case modules. For example, in the area of Software Requirements Specification the project currently has the following material developed or in progress:

- Customer Need Statement
- Needs Assessment Case Module
- Several versions of the DH SRS (pre and post inspection)
- SRS Inspection Case Module, with a package of inspection support forms and tools.
- Requirements Change Case Module
- Use Case Model
- Use Case Inspection Case Module
- Use Case Modification Case Module
- Operational Profile Case Module

We have also developed a number of case modules that are not associated with a particular life-cycle phase. For instance, there is a case module on "Team Problems", which could be used at almost any place in a curriculum. In this case module, the class is divided into teams and each team is provided with a vignette describing a problem associated with the DH team. The titles of the vignettes might give some insight into their make-up: "Slacker on the loose ", "Don't worry your pretty little head", "It's time to make some changes", and "Double Trouble". The teams meet, discuss the problem posed, and answer questions about the team problem. Then, they decide on various approaches that might be taken to solve the problem. Each team summarizes its discussion and conclusions in a team report, which is presented to the entire class.

Table 5: A Digital Home Curriculum Framework

Case Study Area	Course(s)*
Project Planning	SE323 Software Project Management
Software Process Description	SE324 Software Process and Management
Software Requirements Specification	SE322 Software Requirements Analysis
User Interface Specification	SE212 Software Engineering Approach to HCI
Traceability Matrix	SE322 Software Requirements Analysis
Architectural Specification	SE311 Software Design and Architecture
Module Specification	SE311 Software Design and Architecture
Algorithm Design Specification	SE211 Software Construction
Coding	SE211 Software Construction
Unit Test Planning	SE211 Software Construction
Integration Planning	SE311 Software Design and Architecture
System Test Planning	SE321 Software Quality Assurance and Testing
All of the Above	SE101 Introduction to Software Engineering SE400 Software Engineering Capstone Project
* Course number and names are from [10]	

The authors have used the case modules in a variety of computing courses and workshops: undergraduate and graduate courses in software engineering, a workshop in software reliability for faculty and researchers, and a faculty workshop in case study teaching.

6. CONCLUSIONS

Although there is still much work to be done to complete the DH case study, we hope the material developed and our experiences with it provide insight into the value of such an approach. We believe the content, organization, and spirit of the DigitalHome Case Study provides a model for the development of other engineering life-cycle case studies. As a minimum, the DigitalHome can easily serve as a baseline for case studies/modules for Computer Engineering, Electrical Engineering, and System Engineering. Hopefully, this discussion will broaden the reach of this work into other fields of engineering with a goal of building a community of collaborators that will contribute more significantly to the development of case studies, artifacts, and exercises.

7. ACKNOWLEDGMENT

Initial work on the DigitalHome case study was funded as part of the NSF project: "The Network Community for Software Engineering Education" (SWENET) (NSF 0080502). In addition, the current work on the case study is funded through NSF's (DUE- 0941768) "Curriculum-wide Software Development Case Study".

8. REFERENCES

[1] H. Passow, "Which ABET Competencies Do Engineering Graduates Find Most Important in their Work?", **Journal of Engineering Education**, Vol. 101, No. 1, January 2012, pp. 95-118.

[2] **Criteria For Accrediting Engineering Programs, 2011-2012 Accreditation Cycle**, Engineering Accreditation Commission, ABET Inc., October 30, 2010.

[3] G. Friedman, A. Sage, "Case Studies of Systems Engineering and Management in Systems Acquisition", Systems Engineering, Vol. 7, No. 1, January 2004, pp. 84-96.

[4] C. Herreid, "Case Studies in Science: A Novel Method of Science Education", **Journal of College Science Teaching,** February 1994, pp. 221-229.

[5] C. Davis and E. Wilcock, "Teaching Materials Using Case Studies", **UK Centre for Materials Edcuation**, http://www.materials.ac.uk/guides/casestudies.asp, accessed February 2012.

[6] S. Salamah, M. Towhidnejad, T. Hilburn, Developing Case Modules for Teaching Software Engineering and Computer Science Concepts, **Proceedings of 2011 Frontiers in Education Conference**, October 2011.

[7] T. Hilburn, M. Towhidnejad, S. Salamah, "Read Before You Write", **Proceedings of the 24th IEEE-CS Conference on Software Engineering Education and Training**, May, 2011.

[8] M. Fagan, "Design and Code Inspections to Reduce Errors in Program Development", **IBM Systems Journal**, Vol. 15, No. 3, 1976, pp. 258-287, pp. 744-751.

[9] Watts S., Humphrey, **Introduction to the Team Software Process,** Addison-Wesley, 2000.

[10] J. Diaz-Herrera, T. Hilburn (Editors), **Software Engineering 2004:** Curriculum **Guidelines for Undergraduate Degree Programs in Software Engineering, Computing Curriculum Series**, Association for Computing Machinery (ACM) & IEEE Computer Society (IEEE-CS), 2004.