122nd ASEE Annual Conference & Exposition
June 14 - 17, 2015
Seattle, WA
Making Value for Society

Paper ID #11164

Case Study Based Educational Tools for Teaching Software V&V Course at Undergraduate Level

Dr. Priya Manohar, Robert Morris University

Dr. Priyadarshan (Priya) Manohar Dr. Priyadarshan Manohar is an Associate Professor of Engineering and Co-Director Research and Outreach Center (ROC) at Robert Morris University, Pittsburgh, PA. He has a Ph. D. in Materials Engineering (1998) and Graduate Diploma in Computer Science (1999) from University of Wollongong, Australia and holds Bachelor of Engineering (Metallurgical Engineering) degree from Pune University, India (1985). He has worked as a post-doctoral fellow at Carnegie Mellon University, Pittsburgh (2001 – 2003) and BHP Institute for Steel Processing and Products, Australia (1998 – 2001). Dr. Manohar held the position of Chief Materials Scientist at Modern Industries, Pittsburgh (2003 – 2004) and Assistant Manager (Metallurgy Group), Engineering Research Center, Telco, India (1985 – 1993). He has published over 55 papers in peer-reviewed journals and conferences including a 2007 Best Paper Award by the Manufacturing Division of American Society for Engineering Education (ASEE), three review papers and three book chapters. He has participated in numerous national and international conferences. He is a member of ASM International, TMS, ACerS, AIST, ASEE, and a registered Chartered Professional Engineer. Dr. Manohar's research interests include mathematical and computer modeling of materials behavior, thermo-mechanical processing of steels and other metallic materials, microstructural characterization, and structure – property relationships. He has conducted a number of technical failure investigations, consulted on various materials-related problems, and acted as an expert witness in the Court of Law. Dr. Manohar is the immediate past chair of the Manufacturing Division of ASEE and the current Chair of ASM Pittsburgh Chapter.

Dr. Sushil Acharya, Robert Morris University

Acharya joined Robert Morris University in Spring 2005 after serving 15 years in the Software Industry. His teaching involvement and research interest are in the area of Software Engineering education, Software Verification & Validation, Data Mining, Neural Networks, and Enterprise Resource Planning. He also has interest in Learning Objectives based Education Material Design and Development. Acharya is a co-author of "Discrete Mathematics Applications for Information Systems Professionals- 2nd Ed., Prentice Hall". He is a member of Nepal Engineering Association and is also a member of ASEE, and ACM. Acharya was the Principal Investigator of the 2007 HP grant for Higher Education at RMU. In 2013 Acharya received a National Science Foundation (NSF) Grant for developing course materials through an industry-academia partnership in the area of Software Verification and Validation. Acharya is also the Director of Research and Grants at RMU.

Prof. Peter Y Wu, Robert Morris University

Peter Y. Wu is professor of Computer and Information Systems at Robert Morris University. He earned Ph.D. in Computer System Engineering from Rensselaer Polytechnic Institute. He worked for IBM Research Division, first as a post-doc research fellow and subsequently a technical staff member at the T.J. Watson Research Center. He was the chief software engineer and a founding partner of UJB Solutions, LLC, a consulting company in production planning, for two years. He previously held faculty appointments at Hong Kong Polytechnic University, and the University of Pittsburgh. His current research interests are in software engineering, and geographic information systems.

Dr. Ali A. Ansari, Virginia State University
Dr. Walter W Schilling Jr., Milwaukee School of Engineering

Walter Schilling is an Associate Professor in the Software Engineering program at the Milwaukee School of Engineering in Milwaukee, Wisconsin. He received his B.S.E.E. from Ohio Northern University and M.S. and Ph.D. from the University of Toledo. He worked for Ford Motor Company and Visteon as an Embedded Software Engineer for several years prior to returning for doctoral work. He has spent time at NASA Glenn Research Center in Cleveland, Ohio, and consulted for multiple embedded systems

companies in the Midwest. In addition to one U.S. patent, Schilling has numerous publications in refereed international conferences and other journals. He received the Ohio Space Grant Consortium Doctoral Fellowship and has received awards from the IEEE Southeastern Michigan and IEEE Toledo Sections. He is a member of IEEE, IEEE Computer Society and ASEE. At MSOE, he coordinates courses in software quality assurance, software verification, software engineering practices, real time systems, and operating systems, as well as teaching embedded systems software.

Case Study Based Educational Tools for Teaching Software V&V Course at Undergraduate Level

Abstract:

One of the critical problems facing software industry today is the lack of appreciation of the full benefits that can be derived from Software Verification and Validation (V&V) and an associated problem of shortage of adequately trained V&V practitioners. To address this situation, software V&V course curriculum is being improved at author's institution via a NSF-funded project. The basic objectives of this project are to improve software education to make it well aligned with academic research as well as industry best practices. In addition, it is aimed that the course material could also be used to enhance on-the-job professional training in SW industry settings, thereby helping to increase the pool of professionals with contemporary V&V knowledge and skills.

The new course curriculum enhancement described in this paper is guided by academic research and industry best practices that focus on four specific V&V focus areas*: requirements engineering, reviews, configuration management,* and *testing*. Among many educational tools that are being developed to achieve project objectives, the work related to the development of case studies is described here. Historically, case studies have been as educational tools in business, law and medicine but not so much in software engineering. The hypothesis is that case studies would be effective educational tools to introduce real-world professional practices into the classroom which would help the students in identifying and solving problems, and develop a perspective on knowledge application. In this paper we describe a set of V&V related case-studies that we have drawn from industry experiences and developed them as pedagogical tools. These case-studies cover several important topics in S/W V&V domain such as software testing, legal issues in software, software consumer protection, and requirements from the customers' perspectives. We will also report on the results of initial implementation of case studies related to software testing in the classroom.

Case Study Based Educational Tools for Teaching Software V&V Course at Undergraduate Level

1. Introduction – Case Study Based Education

Engineering education must strike a balance between the knowledge of theoretical concepts and developing an ability to apply the theory to solve real world problems. Such a balance between theory and practice requires careful handling of two types of knowledge called *episteme* by Aristotle (meaning theoretical knowledge) and *phronesis* referring to practical knowledge [1]. It is the practical knowledge that is interesting to the student and immediately useful to the community. However, it has been realized that the practical knowledge cannot be easily taught in a class room setting as it requires lots of time and experience. The use of case studies is therefore important because it taps into practical knowledge and real world experiences that students are able to relate to and learn from. Among many other educational tools that have been developed to realize the learning objectives in computing field, tools based on case studies are definitely in short supply [2]. Traditionally, case studies have been used as educational tools in areas of business, law, ethics, economics, systems engineering and medicine but not so much common in software engineering. Case studies have many unique and distinct advantages in software engineering education including:

- Application of knowledge or skills in a real-world setting
- Identification and clearer definition of the problem
- Participative, collaborative, inclusive and team based approach
- Opportunities for creative brain storming, and
- Developing solution of a problem.

Therefore, the hypothesis presented in this paper is that the case studies would be effective educational tools to introduce real-world professional practices into the classroom which would help the students not only in honing their problem solving skills but also self-directed learning skills and team skills. As the case studies are grounded in real world, they would offer opportunities to develop a perspective on knowledge application, project management and project economics in addition to the domain knowledge enhancement in software engineering.

The sources for case studies can be diverse. For example, at the University of Waterloo, Canada [3], the researchers have reported gathering data and information for developing case studies *via* undergraduate student project work, co-op experiences / summer work, senior year capstone projects, industry partners and graduate (ME/MS) thesis projects. In the present work, the case studies have been drawn mainly from industrial partners and large scale government projects as well as from the direct professional consulting experience of the authors. These sources for case study materials thus maintain the currency of knowledge and therefore would be more useful to the students as they prepare for their own working careers.

Another important and emerging aspect of engineering education is the fully on-line engineering degree programs. While there are just a few completely online undergraduate programs available at this time, more and more classes are going online to facilitate the education of nontraditional students such as mid-career employees and military personnel [4]. In order that the online education is at least equally effective (if not better) than face-to-face education in traditional classroom in all aspects such as academic quality, rigor and outcomes, appropriate teaching tools must be developed to suit the online teaching / learning media. In this regard, we believe the case study based education is one of the superior tools to deliver an equivalent laboratory experience for the online students!

The process for developing case studies in described in section 2, a fully developed case study in the domain of software testing is presented in Section 3, the instructions and teaching notes are given in Section 4, pedagogy and educational outcomes are discussed in Section 5 while example of the implementation of software testing case studies is given in Section 6 in this paper.

2. Template for Case Study Development in SW Testing

Testing is an investigative process in which a software system or component is evaluated against a set of predefined inputs to observe whether or not it gives the expected results. If the results are met then the user requirements are met. Testing results in software quality and reliability and helps product development by identifying errors or missing requirements [5]. Whether one is writing an individual unit test or designing a product's test plan, it is important to take a step back and think about how effective the tests are at detecting and reporting bugs in the code.

2.1 Basics Ideas to be introduced in Software Testing Education

For professional software technology companies software testing is a critical aspect of reliability and quality of their business. The basic purpose of testing is to validate the testing coverage of the application being evaluated. Writing effective test cases is a skill that can be achieved by some experience and in-depth study of the application on which test cases are being written.
For example, Google [6] suggests several important qualities that every test should try to maximize:

- **Fidelity**: When the code under test is broken, the test fails. A high fidelity test is one which is very sensitive to defects in the code under test, helping to prevent bugs from creeping into the code. Maximize fidelity by ensuring that tests cover all the paths through the code and include all relevant assertions on the expected state.
- **Resilience**: A test shouldn't fail if the code under test isn't defective. A resilient test is one that only fails when a breaking change is made to the code under test. Maximize resilience by only testing the exposed application programming interface (API) of the code under test and avoid reaching into internals. Favor stubs and fakes over mocks. Don't verify interactions with dependencies unless it is that interaction that you are explicitly validating. A flaky test obviously has very low resilience.
- **Precision**: When a test fails, a high precision test tells you exactly where the defect lies. A well written unit test can tell exactly which line of code is at fault. Poorly written tests (especially large end-to-end tests) often exhibit very low precision, indicating that *something* is broken but not *where*. Maximize precision by keeping tests small and tightly focused. Choose descriptive method names that convey exactly what the test is validating. For system integration tests, validate state at every boundary.

These three qualities are often in tension with each other. It's easy to write a highly resilient test (the empty test, for example), but writing a test that is both highly resilient and high fidelity is hard to do. The students begin to learn how to do software testing by creating tests scripts. The development of test scripts therefore needs greater attention and a systematic procedure as suggested below.

2.2 Test Case Development Template

An example of test case development template is given in Figure 1 below.

<Client> Test Case Name			
Project ID	//the unique project identifier		
AUT Name	//the definitive name of the Application Under Test (AUT)	**AUT Version**	//the definitive version information for the Application Under Test (AUT)
Iteration ID	//the unique identifier for the iteration this test is being conducted in	**Date of Test**	//MM/DD/YYYY
Test ID	//the unique identifier for the test		
Purpose of Test	//a brief description of the purpose of the test including a reference where appropriate to the requirement that is to be tested (consider providing references to the requirements specification, design specification, user guide, operations guide and/or installation guide), as well as any dependencies from or to other Test Cases		
Test Environment	//a brief description of the environment under which the test is to be conducted (may include a description of the state of the AUT at the start of this test, details regarding the platform or operating system, as well as specific information about data used in this test)		
Test Steps	//concise, accurate and unambiguous instructions describing the precise steps the Tester must take to execute the test, including navigation through the AUT as well as any inputs and outputs		
Expected Result	//a brief and unambiguous description of the expected result for passing of a test subsequent to test execution		
Likely Problems / Bugs Revealed	//likely outcomes of testing such as feature not working, expected results not observed, missing or inaccessible features (optional field)		

Figure 1. Test case development template.

The benefits of developing the template for a case study are twofold: (a) template provides a standardized way to document the background information, description and objectives of case studies and (b) facilitates identification of any missing information or gaps of knowledge for the students as they attempt to solve the questions based on the case study. This allows improving the description contained in the case study as it is delivered in the following iterations in the future classes.

The developed case studies can be integrated in the curriculum either as home work assignments or as in-class exercises. The authors have utilized the case studies in both ways and it is suggested here that the more complex case studies should be assigned as home works while relatively narrow and focused case studies may be discussed as in-class exercises. It should be noted here that appropriate theoretical framework needs to be established via lectures to lay the foundation before case studies based on the relevant topics are given to the students. In class quizzes are then utilized to assess the effectiveness of student learning where students gain the necessary theoretical framework via lectures and then were able to understand the practical applications of the principles via case studies.

Student performance in case studies can be assed based on their in-class participation and the accuracy of their response to the case study questions. A grading rubric may be developed appropriate for each case study depending upon whether it is an individual or group assignment and if it is an in-class or home work assignment.

2.3 Applications of Test Case Development Template

The students will be expected to write down such template for any piece of software assigned to them for testing. For example, a case for credit card transactions testing is given as follows that students can work on to develop test cases to test the functionality of the credit cards:

Basic Test Cases for Credit Card Transactions:

Case 1: Check for invalid Characters in Credit Card.
Description: Enter invalid characters @@@@34534"asd".
Expected Result: Error message should appear informing that invalid value is entered.
Case 2: Check for wrong Credit Card type.
Description: Enter invalid Credit Card type e.g. Enter AmEx in place of VISA.
Expected Result: Error message should appear informing that invalid Credit Card is entered.
Case 3: Check for wrong Expiry Date.
Description: Select wrong month and year of expiry date.
Expected Result: Error message should appear informing that invalid Expiry date has been entered.
Case 4: Check for CVV number with the invalid characters as well as with the alphabetic and alpha numeric values.
Description: Enter invalid CVV number. Like: ABC or a3c or @@" or "1".
Expected Result: Error message should appear information. Invalid characters are entered.
Case 5: Check for validation messages while entering wrong billing information.
Description: Check for Maximum and Minimum value acceptance. Check for invalid Characters. Check for Numeric value acceptance where numeric values are required.
Expected Result: Error message should appear while entering invalid values.

3. A Sample Case Study for SW Testing

Based on the procedures given in Section 2 above, the test cases can then be bundled together for developing a full software testing case study. To maintain consistency and to standardize how a case study is prepared and presented to the students, a template has been developed in this work. For example, the template for software testing case study is shown in Figure 2 given below.

Focus Area :	**Testing**
Case Module Name :	*Industry Test Case Development*
Prerequisite Knowledge:	Before attempting this module you need to have knowledge of: 1. Software Testing, Test Outline and Test Case Development
Learning Outcomes/Objectives:	Upon completion of this module you be able to meet the following **ABET *Criterion 3* Outcomes:** 1. **Outcome *b***: *An ability to design and conduct experiments, as well as to analyze and interpret data..* 2. **Outcome *g***: *Graduates have an ability to communicate effectively*. 3. **Outcome *k***: *Graduates have an ability to use the techniques, skills, and modern engineering tools necessary for engineering practice.*
Keywords:	Test Plan, Requirements, Formal review, informal review, reviewer, Peer review, recorder, Moderator, Author
<u>Scenario</u>:	OptSoft Inc. has been awarded a project to develop a Hospital Management System. This system has the following subsystems: • Admit-Discharge-Transfer (ADT) / Patient Registration System (PRS) • Hospital Information System (HIS) • Radiology Information System (RIS) • Picture Archiving and Communication System (PACS) • Image Acquisition Modality The Software Requirements Specification (SRS) for the System has already been completed and accepted by the customer. Your QA Manager has asked your team to come up with test cases for the PRS that will be used to test the PRS features. Everyone on the team should be able to answer the following: • What am I going to test? • What's the importance of having a formal template for test cases development? • What is the importance of preconditions in test cases development?

Figure 2. A template for case study development in software testing domain.

The case studies are then integrated in to the curriculum as in-class exercises or as home work assignments. The case study tasks may be accomplished by the students working individually or as student teams consisting of 2-4 students per team. The case study contains scenarios or real world stories along with the background material such as setting, personalities, sequence of events, relevant data, problems and conflicts. Students involved in group work related to case studies can develop skills required for success such as group decision making, consensus building, negotiation and tolerating differences of opinion within a diversified work place [7].

It is noted here that the template presented in Fig. 2 does not include a field that explicitly defined what kind of problems or bugs the students are expected to detect in a given scenario that was shown as an optional field in the template presented in Figure 1. This has been done deliberately in this case to avoid testing bias and not to limit student efforts to detect a certain type of bug. A piece of software may have several types of bugs like syntactical errors, data/file update errors, input/output errors, invalid data entry errors and so on. The students are expected to do the software testing based on what the SW is supposed to do, obtain test results and then make their own decisions as to the type of bug or bugs that they found during software testing.

4. Case Study Teaching Notes

Instructors may assign additional exercises suggested as follows:

Exercise:	■ Develop ten test cases that are relevant to the application you are testing such as those described in Section 2.3. ■ Use the "Test Case Template" given by your instructor to write scripts for the ten test cases that you just developed (example template for writing test scripts is provided in Figure 1 in this paper) ■ Identify test cases that can be automated. (Assume appropriate information when necessary.)

Instruction and Assessment Procedure:

Instructing Notes:	This is to be delivered as a classroom/Homework assignment. <u>Class 1 (50 minutes)</u> 1. Ask the students to individually prepare the test cases 2. Form teams of 4 students in class 3. Ask the teams to discuss the questions given in the case study. 4. Discuss as a class (10 minutes)
Assessment Procedure:	In-class short quiz for this module may be developed and given to the students in addition to their performance in the test cases, test script writing and case review

5. Educational Outcomes Assessment

The case study based is broad in terms of its effectiveness in educational outcomes and it has been suggested that it can be used to deliver all eleven '***a***' through '***k***' criteria of ABET accreditation [2]. The flexibility of case studies coupled with the richness of data and information analysis, decision making education and conflict resolution results in strong links with ABET criteria. Kauffman et al [7] have mapped case study outcomes to the ABET criteria for engineering economy case studies. Such analysis is adopted here for case studies in software engineering as shown in Table 1.

Table 1. Case study analysis in software engineering and its relation to ABET criteria.

ABET Criterion	Software Engineering Case Study Analysis
(b) An ability to design and conduct experiments as well as to analyze and interpret data	Case studies requires students to find or develop the important information and ignore data that is not relevant
(c) an ability to design a system, component or a process to meet desired needs	Case studies requires students to confront complex issues such as trade off analysis along with time, resource and risk management decisions
(d) an ability to function on multi-disciplinary teams	Case studies requires students to solve case problems, they must also learn to negotiate and understand different viewpoints prior to their decision making
(e) An ability to identify, formulate, and solve engineering problems	Case studies requires students to identify important data and ignore irrelevant data, actively look for missing data or make appropriate assumption and use mathematical / computer simulation based tools to solve engineering problems
(g) an ability to communicate effectively	Case studies requires students to make presentation of case analysis results in both oral and written formats
(h) the broad education necessary to understand the impact of engineering solutions in a societal and global context	Critical thinking required by case study analysis promotes systems thinking related to larger impact of decision alternatives
(k) an ability to use the techniques, skills, and modern engineering tools necessary for engineering practice	Case studies requires students to learn and apply contemporary engineering tools to solve case problems

Pedagogical outcomes that are relevant for software verification and validation have been identified at the author's institution based on ABET Criterion 3 outcomes assessment. The relationships between the specified ABET outcomes for this course and their correspondence with the revised Bloom's taxonomy for STEM disciplines is shown in Table 2. The seven levels (taxa) of conceptual and procedural knowledge and skills taxonomy proposed by Girgis [8] mentioned in Table 2 are defined as follows:

Taxa I - Pre-knowledge Conceptual Experiences: hands-on laboratory experiences via demonstrations, physical models, practical applications to demonstrate, visualize and observe basic concepts

Taxa II - Basic Conceptual Knowledge: learning, understanding, memorizing basic engineering concepts, definitions, terms, symbols, theories, laws and equations

Taxa III - Applied Conceptual Knowledge: solving simple concept-based problems and conducting related laboratory experiments

Taxa IV - Procedural Knowledge: working knowledge of solving multi-concept engineering problems

Taxa V - Advanced Knowledge and Analytical Skills: inter-domain and open-ended problem solving skills

Taxa VI - Project-based Knowledge: creative, conceptual, analytical, design, manufacturing and management skills

Taxa VII - Professional Engineering Knowledge and Practices: life-long learning experiences, skills and practices

Table 2. Expected pedagogical outcomes for software V&V course at author's institution.

Applicable ABET Criterion 3 Learning Outcomes for Software V&V course at author's institution	**Conceptual and procedural knowledge taxonomy based on revised Bloom's taxonomy for STEM Disciplines [8, 9]**
b. An ability to design and conduct experiments, and analyze and interpret data	I & III
c. an ability to design a system, component or a process to meet desired needs	IV, V & VI
e. An ability to identify, formulate, and solve engineering problems	II, IV & V
f. An understanding of professional and ethical responsibilities	V & VII
g. An ability to communicate effectively	III, IV & V
h. Broad education necessary to understand the impact of engineering solutions in a global and societal context	VI
i. Recognition of the need for and an ability to engage in life-long learning.	VII
j. A knowledge of contemporary issues	V & VI
k. An ability to use the techniques, skills, and modern engineering tools necessary for engineering practice	VI & VII

It is clear from the information presented in Tables 1 and 2 that it is possible to evaluate student learning outcomes ***b, c, e, f, g, h*** and ***k*** using the case study based educational tools.

6. Implementation of the Case Study Method

One of the authors has been delivering this class since 2005. The student performance in each assessment task was measured regrouped in terms of ABET outcomes to work out percentage of students that scored within certain levels of assessment vector as detailed in Table 3 given overleaf.

Table 3: Descriptors of ABET Outcomes Assessment Vector

% of students with at least 80% or better score in assessment tasks	Descriptor of the Resulting Status
90% – 100%	Excellent (E)
80% – 89%	Proficient (P)
70% – 79%	Adequate (A)
60% – 69%	Concern (C)
< 60%	Weakness (W)

These are the percentage of students scoring 80% or better in assessment tasks. The bar graph depicting this analysis is shown in Figure 1 and descriptive explanation of the assessment vector is given in Table 3. The detailed class performance in ABET Criterion 3 Outcomes assessment is shown in Figure 3 below.

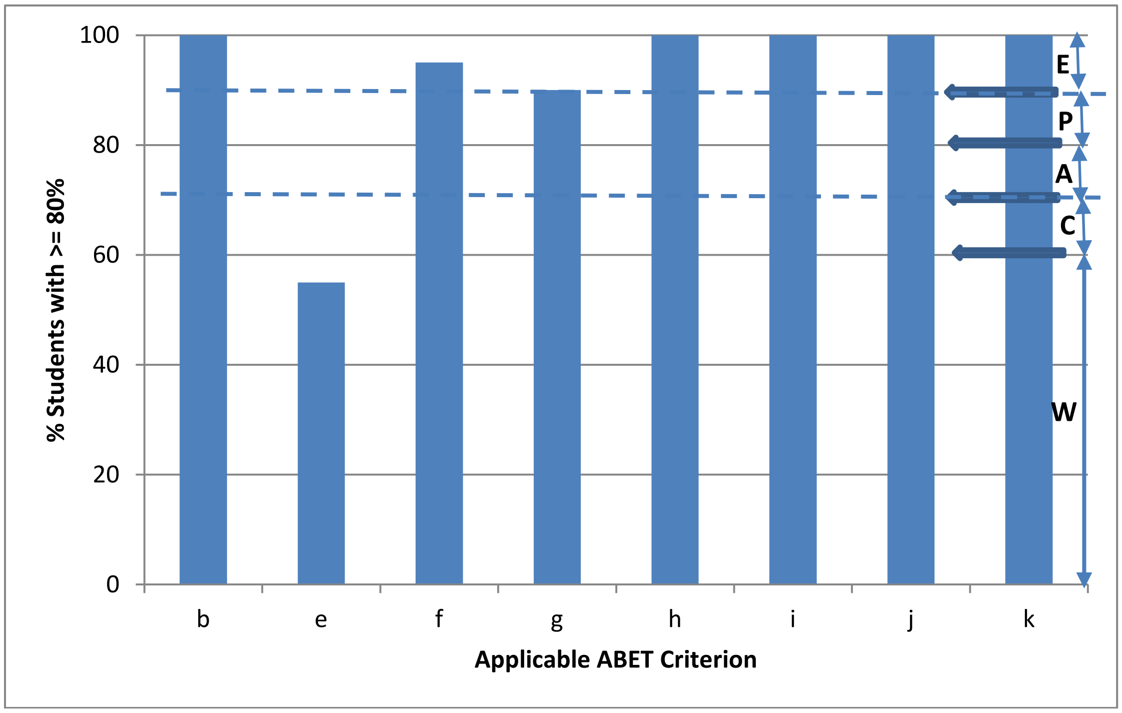

Figure 3: Student outcomes assessment with respect to the specified ABET criteria. (E – Excellent, P – Proficient, A – Adequate, C – Concern and W – Weakness)

This class assessment was performed in Spring 2013 iteration when case studies were not available at that time. It is seen from Figure 3 that there was a weakness in learning outcome ***e*** (an ability to identify, formulate, and solve engineering problems) where less than 60% of the students scored better than 80% in the assessment tasks. One of the main reasons for the lower outcome percentage was because the student performance data was obtained through exams and some student did not perform well in the exam environment. Thus exams may not be the best suited tools for assessing Outcome (***e***). It is shown in Table 1 that case study based education can be used to enhance outcome ***e***. Thus the case studies will be delivered in the upcoming iteration of this course in Spring 2015 and outcome assessment in this ABET criteria will be evaluated based on case study based evaluation tasks. The results of this evaluation will be presented in the conference in June 2015. Progressively case study based educational tools will be delivered in several aspects of software V&V area such as legal issues in software, software consumer protection, and requirements from the customers' perspectives. The results of this implementation will be reported at later conferences.

7. Summary and Future Work

To develop case studies, the needed data, information and scenarios have been drawn from industry and from professional consulting work of the authors. The template for developing case studies has been created and presented in this paper. Case notes and instructors supplementary materials have been prepared. The case studies developed here cover many areas in software verification and validation knowledge domain including software testing, legal issues in software, software consumer protection, and requirements from the customers' perspectives. Applicable student learning outcomes for the software V&V course have been determined and their relationship to ABET criteria and revised Bloom's taxonomy for STEM disciplines has been mapped. The effectiveness of case study based educational tools has been determined based on this evaluation context. The baseline for evaluation of new educational tools has been established as student learning outcomes assessment for Spring 2013 iteration when case studies were not available. The newly developed case studies were introduced to the students in Spring 2015 term. The effectiveness of these teaching / learning tools will be assessed and the results will be presented in the conference.

Acknowledgement

The authors acknowledge the support of NSF through a grant entitled "Collaborative Education: Building a Skilled V&V Community", NSF-TUES Award # 1245036. In addition, the input of industrial partners for this project including Eaton Electrical Corporation, PNC Bank, ServiceLink, and JDA Software Group is gratefully acknowledged.

References

[1] Bertha C.: “How to teach engineering ethics course with case studies”, proc. ASEE Annual Conference, 2010
[2] Towhidnejad M., Hilburn T. B. and Salamah S.: “Reporting on the use of a software development case study in computing curricula”, proc. ASEE Annual Conference, 2011
[3] Nespoli O. and Lambert S.: “Engineering design case studies: effective and sustainable development methods”, proc. ASEE Annual Conference, 2010
[4] Fisher F., Hadim H., Esche S., Ubell R. and Chassapis C.: “Feasibility of a fully online undergraduate mechanical engineering degree for non-traditional learners”, proc. ASEE Annual Conference, 2007
[5] Acharya S. and Manohar P.: “Cases in Software Verification and Validation”, a textbook under publication by Alexander Street Press, 2015.
[6] Google: “*Effective Testing*”, **http://googletesting.blogspot.com** downloaded 12/29/2014
[7] Kauffman P., Abdel-Salem T., Williamson K. and Considine C.: “Privatization initiatives: a source for engineering economy case studies”, proc. ASEE Annual Conference, 2005
[8] Girgis, M: “A new engineering taxonomy for assessing conceptual and problem-solving competencies”, proc. ASEE Annual Conference, 2010.
[9] Bloom, B. S.: “Taxonomy of Educational Objectives”, 1956, Boston, MA