AC 2008-639: WORLD-CLASS OUTCOMES ASSESSMENT ON A SHOESTRING

Joseph Clifton, University of Wisconsin-Platteville

Joseph M. Clifton is a Professor in the Department of Computer Science and Software Engineering at the University of Wisconsin – Platteville. He has a Ph.D. from Iowa State University. His interests include software engineering, real-time embedded systems, and software engineering education.

Rob Hasker, University of Wisconsin-Platteville

Robert W. Hasker is a Professor in the Department of Computer Science and Software Engineering at the University of Wisconsin-Platteville. He has a Ph.D. from the University of Illinois at Urbana-Champaign. His interests include software engineering education, programming languages for introductory courses, and formal specifications.

Mike Rowe, University of Wisconsin-Platteville

Michael C. Rowe is an Associate Professor in the Department of Computer Science and Software Engineering at the University of Wisconsin - Platteville. He has a Ph.D. from the University of North Texas. His interests include software engineering, software quality assurance techniques, student projects, and software engineering education.

World-Class Outcomes Assessment on a Shoestring

Abstract

In the fall of 2004, the software engineering faculty at the University of Wisconsin - Platteville developed a set of outcomes and assessment procedures using the principles presented at an ABET Faculty 2.0 Workshop. In addition, we have taken extra steps to improve the quality of the assessment process and reduce the effort required. The result is a tightly specified assessment process that allows us to achieve quality assessment results at a reasonable expenditure of faculty time and effort.

Introduction

Criterion three of the ABET engineering accreditation process states that program outcomes must be assessed with evidence that the results of this assessment process are applied to the further development of the program.[3] Anecdotally, many who go through the ABET accreditation process view this criterion as the most problematic. Moreover, satisfying this criterion usually requires significant ongoing efforts.

In the fall of 2004, the software engineering faculty at the University of Wisconsin - Platteville developed a set of outcomes and assessment procedures using the principles presented at an ABET Faculty 2.0 Workshop.[4] These outcomes assessment procedures share a number of common practices delineated by other authors who have chronicled their experiences with ABET outcomes assessment.[2,4,6] Our particular instantiation has:

- Seven program outcomes with two to five performance criteria established for each outcome.
- Two to four measurements for each performance criterion, with at least one direct and one indirect measurement for each. The direct measurements consist of in-course assessments and direct observation. The indirect measurements consist of course surveys and graduating senior exit surveys.
- A fixed set of rubrics for each of the measurements
- Semester assessment reports summarizing assessment data, identifying problem areas, suggesting improvements, noting where changes due to assessment lead to improvements, and suggesting changes to the assessment process

In addition to the above, we have taken a few extra steps to improve the quality of the assessment process and reduce the effort required. We have specified how each measurement will be performed. This specification helps ensure that a given measurement is performed reliably regardless of the semester or instructor. We have normalized each rubric to produce a numerical result from one to five for each student. We have created automated trigger values based on three sets of criteria that flag those measurements that are below desired levels and therefore require further analysis. These steps go beyond those discussed by other authors. [2,4,6]

The stricter specifications make the results uniform from semester to semester but have the added benefit of reducing the time commitment of the faculty. Furthermore, they allow mechanizing much of the process for tabulation and problem identification. This also simplifies further analysis of identified problems and suggestions for improvement.

We have gone through several cycles of measurement, change, and re-measurement. There are some curriculum areas where significant improvement has been made due to this process. We also have a couple of areas that continue to indicate the existence of problems and we have focused our persistent attention on their remediation. In any case, this tightly specified assessment process allows us to achieve quality assessment results at a reasonable expenditure of faculty time and effort.

Note that this paper is not advocating the position that all program changes should be tied to outcomes assessment. The focus of the paper is on outcomes assessment as mandated by ABET, but we acknowledge other sources are also very important. In particular, our program also receives input from a program advisory board, a college advisory board, three and five year surveys of graduates and our graduates' managers, and benchmarks against other programs.

Outcomes

The current software engineering outcomes, adopted in December of 2004, are:

A. Foundation: Graduates shall have a strong foundation in science, mathematics, and engineering, and can apply this fundamental knowledge to software engineering tasks.

B. Development: Graduates can effectively apply software engineering practice over the entire system lifecycle. This includes requirements engineering, analysis, prototyping, design, implementation, testing, maintenance activities and management of risks involved in software and embedded systems.

C. Process: Graduates know various classical and evolving software engineering methods, can select appropriate methods for projects and development teams, and can refine and apply them to achieve project goals.

D. Professionalism: Graduates are knowledgeable of the ethics, professionalism, and cultural diversity in the work environment.

E. Quality: Graduates can apply basic software quality assurance practices to ensure that software design, development, and maintenance meets or exceeds applicable standards.

F. Presentation: Graduates have effective written and oral communication skills. Graduates can prepare and publish the necessary documents required throughout the project lifecycle. Graduates can effectively contribute to project discussions, presentations, and reviews.

G. Growth: Graduates understand the need for life-long learning and can readily adapt to new software engineering environments.

Performance Criteria and Measurements

To assess the software engineering outcomes, two to five performance criteria have been established for each outcome. In total, there are 28 performance criteria. For each performance criterion, at least one direct and one indirect measurement are specified. All measurements are done by the software engineering faculty members with the exception of one measurement done by the Philosophy faculty for an ethics assessment. The methods used for assessment fall into the following categories:

> **In-Course Assessment**: A specific item such as a test problem or quiz problem is assessed independently of the grade for the test or quiz. The assessment is done using a separate rubric with the purpose of providing a direct measurement of a specific performance criterion.
>
> **Direct Observation**: The student is observed while performing some task such as giving a presentation, participating in a code inspection, etc. The assessment is done using a rubric with the purpose of providing a direct measurement of a specific performance criterion.
>
> **Course Surveys**: Course learning outcomes are included in each course. Students are surveyed in each course and asked whether the course learning outcomes are being met. A mapping of course outcomes to performance criteria is used. This is an indirect assessment measurement.
>
> **Graduating Senior Exit Surveys**: Graduating seniors provide feedback by completing a standard exit survey during the last week of their final semester. The survey is designed using the performance criteria. This is an indirect assessment measurement.

As an example, the performance criteria and measurements for Development Outcome B are:

B.1: Develops requirements specifications.

- **B.1.a.** Assess portions of requirements specifications written for project by individual students in SE 4330.
- **B.1.b.** Student course assessment surveys for SE 4330.
- **B.1.c.** Graduate exit survey.

B.2: Designs experiment using prototype to address a technical risk.

- **B.2.a**. Locally-developed, project-related exam or quiz questions in SE 4330.
- **B.2.b.** Student course assessment surveys for SE 4330.
- **B.2.c.** Graduate exit survey.

B.3: Produces a system design satisfying requirements.

B.3.a. Locally-developed exam questions in SE 3430.

B.3.b. Assessment of designs developed by groups in SE 4330 for class project.

B.3.c. Student course assessment surveys for SE 3430.

B.3.d. Graduate exit survey.

B.4: Identifies risks in software systems.

B.4.a. Locally-developed exam questions in SE 4330 for which student must identify risks and classify them according to severity.

B.4.b. Student course assessment surveys for SE 4330.

B.4.c. Graduate exit survey.

B.5: Implements software for a real-time embedded system

B.5.a. The instructor shall assess one real-time embedded systems program for each student in SE 4130.

B.5.b. Student course assessment surveys for SE 4130.

B.5.c. Graduate exit survey.

B.6: Maintains an existing software system

B.6.a. Faculty and peer assessment of each student's contribution to the maintenance project in SE 3860.

B.6.b. Student course assessment surveys for SE 3860.

B.6.c. Graduate exit survey.

For each performance criterion, the first measurement listed is always a direct measurement. For a few performance criteria, there are two direct measurements. The direct measurements are done in the upper-division courses to better tie in with the ABET notion that "… program outcomes are statements that describe what students are expected to know and be able to do by the time of graduation."[3] Many are done in our year-long senior capstone project courses, SE 4330 and SE 4730. In some cases, such as assessments for outcome A (Foundation), the students in the capstone courses are told that they will have an in-class assessment, but are not told what the assessment would be. They are told that the assessment will count as a small percentage of their grade, enough so that they take it seriously but not enough that they openly complain. They are also made aware of how important the assessment is to ABET accreditation.

An example of a direct measurement that was used for B.2.b is the following question from the SE 4330 exam in fall 2007:

> An alternative project we discussed was to build a system for checking out laptops (and checking them back in). Recall that the system would support scanning student ids during laptop checkout and computer ids during both check-in and checkout. The system would then use email to notify students of due dates and produce reports so that fines could be issued. Identify likely risks related to this project in each of the following categories, classify the risk by severity (low/medium/high), and describe how you might mitigate that risk: project, technical, business.

The rubric used for the measurement is the following, with 2 and 4 used to cover the cases in between: 1 = poor/no risks, 3 = three moderate risks, 5 = three significant risks.

Other direct measurements are done in obvious courses such as SE 3430, Object-Oriented Analysis and Design; SE 3860, Software Maintenance and Reengineering; and SE 4130, Real-Time Embedded Systems Programming. A number of the direct measurements for the Presentation and Growth outcomes are performed in SE 4110, Senior Seminar, where the student is required to research a topic, then write and present a paper.

For most outcomes, the last two measurements for each performance criterion are indirect. The course assessment surveys list the course outcomes. The students specify how well they believe each outcome was met. For each measurement that uses student course assessment surveys, one course outcome is chosen that most directly maps to what is being measured. For example, measurement B.2.b is tied to course outcome seven in SE 4330:

7. Design and execute an experiment using a prototype to address a technical risk.

The graduate exit surveys are given to software engineering students approximately one week before they graduate. There is a question for each of the 28 performance criteria. The graduating students indicate how well they believe each was achieved. The surveys are anonymous; however, our secretary is responsible for assuring that each graduating student completes the survey. We have had 100% compliance since we started.

Analyzing the Results

The only measurements performed every semester are those done using the graduate exit surveys. The rest of the measurements are done according to an established rotation cycle. When we started, the rotation cycle required that each measurement be performed every time the associated course was taught. After four semesters of data were gathered, the direct measurements, which are certainly harder to perform, were put on a different rotation cycle. Those measurements that often show problems are still assessed every semester the corresponding course is taught; however, other measurements have been put on two-year rotations and a very small number have been put on three-year rotations.

With the exception of course assessment surveys, all measurements are taken for each student. At this time, we cannot guarantee that all students will attend class on the day the course surveys are administered; however, we are considering web-based alternatives to attempt to get 100%. All measurements are normalized to the range 1 - 5, with five meaning “good” and one meaning “poor”. These are then aggregated and are entered into the Assessment Results spreadsheet. A portion of the spreadsheet for fall 2006 is shown below.

	Ave	% 5's	% 4's	% 3's	% 2's	% 1's	5's	4's	3's	2's	1's	Comments
B.1.a	**3.5**	23.5	17.6	47.1	11.8	0.0	4	3	8	2	0	Req - SE 4330 Assess
B.1.b	4.6	56.3	43.8	0.0	0.0	0.0	9	7	0	0	0	SE 4330 Crs Surv - Q5
B.1.c	4.8	75.0	25.0	0.0	0.0	0.0	3	1	0	0	0	Requirements - Exit Survey

B.2.a	**3.4**	29.4	17.6	23.5	17.6	**11.8**	5	3	4	3	2	Prototype - SE 4330 Quest
B.2.b	4.7	68.8	31.3	0.0	0.0	0.0	11	5	0	0	0	SE 4330 Crs Surv - Q7
B.2.c	4.5	50.0	50.0	0.0	0.0	0.0	2	2	0	0	0	Protoype - Exit Survey
B.3.a	4.4	54.8	29.0	16.1	0.0	0.0	17	9	5	0	0	Design - SE 3430 - Quest
B.3.b	4.2	41.2	47.1	0.0	11.8	0.0	7	8	0	2	0	Design - SE 4330 - Assess
B.3.c	4.7	72.2	27.8	0.0	0.0	0.0	13	5	0	0	0	SE 3430 - Crs Surv - Q2
B.3.d	4.8	75.0	25.0	0.0	0.0	0.0	3	1	0	0	0	Design - Exit Survey
B.4.a	**2.9**	11.8	23.5	17.6	**35.3**	**11.8**	2	4	3	6	2	Risks - SE 4330 - Quest
B.4.b	4	25.0	56.3	12.5	6.3	0.0	4	9	2	1	0	SE 4330 - Crs Surv - Q6
B.4.c	4.5	50.0	50.0	0.0	0.0	0.0	2	2	0	0	0	Risks - Exit Survey
B.5.a	---	---	---	---	---	---						RT Soft - SE 4130 - Assess
B.5.b	---	---	---	---	---	---						SE 4130 - Crs Surv - Q4
B.5.c	5	100.0	0.0	0.0	0.0	0.0	4	0	0	0	0	RT Soft - Exit Survey
B.6.a	---	---	---	---	---	---						Mtce - SE 3860 - Assess
B.6.b	---	---	---	---	---	---						SE 3860 - Crs Surv - Q8
B.6.c	4.3	25.0	75.0	0.0	0.0	0.0	1	3	0	0	0	Maintenance - Exit Survey

The comments column indicates how each item was assessed. For example B.1.a, B.2.a and others are direct measures; the comments indicate whether the assessment was based on an assignment, question, or otherwise. B.1.b, B.2.b, and others are measurements based on course surveys given to students. These refer to the specific course outcome from which the measurement is taken. B.1.c, B.2.c, and others are based on surveys given to graduating seniors. Blanks and missing data indicate assessments that were not done in the particular semester. For example, B.5.a was not performed in the fall because it is measured in a spring-only course. The resulting spreadsheet both summarizes all measured results and provides a mechanism to track back to the source data.

In the case of direct measures, the rubrics are recorded to provide a context for the measurements and continuity between semesters and instructors. For example,

> F.4.c. Instructor shall review each student's performance during technical walkthroughs and reviews (SE 4330 and/or SE 4730). For each role of Recorder, Reader, Moderator, and Reviewer, rate each student's performance using the scale:
>
> 1 = poor, 2 = lacking, 3 = okay, 4 = very good, 5 = excellent

The spreadsheet has automatic triggers that highlight, in bold red, measurements that fail to meet certain criteria. These criteria are used to identify those items for which a careful analysis must be performed. The current criteria are

A. The number of responses in the "1's" column is greater than 0. Since the program must strive to assure all students are achieving the outcomes, any nonzero entry in this column needs to be addressed.
B. The "% 2's" column is greater than 20% AND the number of responses in the "2's" column is greater than 1.
C. The average of the item is less than 3.5.

The line for B.4.a in the above table has three indicators showing problems: the overall average, the percentage of "2's", and the percentage of "1's". The careful reader will note an anomaly: B.1.a is highlighted even though it does not meet any of criteria A-C. That is because up through the fall of 2006, criterion C was that "The average of the item is one standard deviation lower than the average of the averages of all items." After analyzing several semesters of data, we found that this criterion generated too many false alarms. We changed criterion C so it is based on an average because this is both simpler and more useful.

Each highlighted measurement is analyzed to determine if there truly is a deficiency or concern that needs to be addressed or if the measurement is an anomaly. If a direct assessment item triggers, the specific assessment question or method is scrutinized and individual students are taken into consideration. For example, if an individual fails the course, their measurement can be disregarded since they will have to repeat the assessment the next time they take the course. If an indirect measurement triggers due to one student's low response or the average being slightly below 3.5, and no other measurement for that performance criteria causes a trigger, usually no changes are recommended and the item is subject to closer review the next measurement cycle. Thus, the assessment measurements are also subjected to validation as suggested by Lindgard.[4]

Closing the Loop

For those measurements for which it has been determined a deficiency or concern exists, curriculum changes are recommended. The recommendations are based on discussions between faculty about why the item triggered and possible ways to improve. An assessment report is generated each semester that summarizes the results, documents the analysis, and outlines recommended changes. At the start of each semester, the SE faculty members agree on which of the recommended changes to adopt. Those recommendations that can be directly implemented by the SE faculty are implemented as soon as appropriate. Those recommendations requiring department or college approval to implement are ushered through the necessary faculty governance channels.

The effectiveness of changes made due to assessment is measured in subsequent assessment cycles. Since the type of measurements to be performed and the rubrics used are so well specified, we can achieve a good indication of the success of any particular set of changes. Thus far, we have had some changes that were quite successful but have a few areas that continue to be problematic.

One example of success is for performance criterion A.2: "Uses mathematical modeling techniques to specify systems." In the fall of 2005, this was flagged as a concern area by failures in three of four different measurements. The faculty agreed upon the following remedies:

- Get better reference materials for use in the SE 4330 class when teaching formal methods
- Get some kind of tool to do Z specifications
- Lengthen the time spent in SE 4330 lectures on formal methods.

While it might seem obvious that these would improve performance, the difficulty of this topic makes improvement uncertain. However, subsequent measures have shown significant improvement in this area. Two faculty members teach this course, approaching the material from very different viewpoints. In spite of this, all students passed direct measures made by both faculty members, one in 2006 and the other in 2007. The process ensured both that measurable progress has been made and that this progress was not dependent on just one faculty member.

An example of continued concern is for performance criteria A.3: Analyzes run-time performance of algorithms. This is assessed in the second-semester capstone course, SE 4730, which is taken at least three semesters after the data structures courses in which the topic is covered. The direct measurement first flagged a problem in spring 2005, with a mean of 1.5. The recommended and implemented change at that time was to "convey the material in SE 2430 and SE 2630 in a more general way so that algorithm efficiency becomes a natural calculation versus a memorization." In spring 2006, none of the SE 4730 students had experienced the change and the measurement was still very low, 1.8. By spring 2007, some of the students in SE 4730 had experienced the change and the measurement rose to 2.6. This is still quite low. Although we are hoping for better results in spring 2008, we will not know until then whether or not other changes will be necessary.

Besides changes to the curriculum, the assessment reports also recommend changes to the Assessment Plan itself. The Assessment Plan is modified almost every semester. As is typical for many software engineering process documents, revision changes are listed within the document itself.

The Shoestring

Faculty at other institutions have expressed a concern that outcomes-based assessment as mandated by ABET could require an inordinate amount of work. One colleague compared it to monitoring the minutiae of every plant in a back-yard tomato patch: stem lengths, water consumption, tomato sizes, etc.[1] So the question is not whether outcomes-based assessment can be done, but whether it can be done on a limited budget.

The software engineering program at the University of Wisconsin - Platteville involves just four full-time faculty in a department of nine. One faculty member is new, so just three of the faculty members did the assessments described here. Building the process certainly took time, but the ongoing costs are quite reasonable. The indirect measures are generally very easy to perform – typically a few minutes of class time for a survey. Many of the direct measures are based on existing exam questions and assignments. For these, the incremental cost consists of converting scores into the 1-5 scale with possible adjustments for non-assessment-related issues (such as late penalties and failures to follow procedures). A few assessments consist of quizzes administered in upper-divisional courses. These do require additional work, but that work is minimized by sharing the quizzes between instructors. Accounting for all of these, we estimate each faculty member averages 4 to 5 hours per semester collecting assessment-related measures. In addition, one faculty member spends another 5 hours combining all of the data, presenting the results to the other faculty, and writing the semester's report.

Twenty hours a semester is not trivial. Faculty have many duties, and could certainly make use of that time in other ways. However, it certainly is a defensible way to spend teaching efforts given the ability to identify issues and track improvements.

Conclusion

Similar to what is reported in reference one below, we can map our ABET assessment process to the CMMI.[2] Where we take this CMMI-like process a few steps further is that we specify what will be measured and how it will be measured. The measurements and rubrics, agreed upon by all of our SE faculty members, are the same regardless of who teaches a given course. This has the advantage of process consistency and produces a baseline from which changes implemented due to assessment triggers can be measured and analyzed at a later time to determine whether desired improvement was actually achieved. It also has a benefit of reducing the amount time and work that the faculty members have to do each semester. A faculty member teaching a given course does not have to determine what to assess and how to assess it for ABET coverage. This institutionalized process is particularly useful for new faculty members. It also ties in well with the CMMI and ISO 9000 notions that processes are specified to enough detail that new people can join the process and immediately be productive.

References

[1] Budny, Dan, *et. al*. (2003), "Assessment: When is Enough, Enough?", *Frontiers in Education*, Denver, CO, November 2003.

[2] Duggins, Sheryl (2006), "Accreditation – Applying CMM to Software Engineering Education", *Proceedings of the 2006 American Society for Engineering Education*, Chicago, IL, June 2006.

[3] Engineering Accreditation Criteria, Online [January 7, 2008]. Available at http://www.abet.org/forms.shtml.

[4] Lingard, Robert (2007), "A Process for the Direct Assessment of Program Learning Outcomes Based on the Principles and Practices of Software Engineering", *Proceedings of the 2007 American Society for Engineering Education*, Honolulu, HI, June 2007.

[5] Rogers, Gloria (2004), ABET Faculty Workshop 2.0, October 26, 2004, Nashville, TN.

[6] Ward, Michael Ward (2007), "Implementing EC2000 – Perspectives from Both Sides of the Assessment Trench", *Proceedings of the 2007 American Society for Engineering Education*, Honolulu, HI, June 2007.