Using the Vocabulary of Software Engineering to Describe ABET Accreditation

Ronald J. Leach
Howard University
rjl@scs.howard.edu

ABSTRACT
In this paper we use some common terminology from the computing field to help explain several concepts in the ABET accreditation process that appear to be rather widely misunderstood in the educational computing community. We show how the familiar terms "software quality metric," "software process metric," "quality assurance," and the "test-debug cycle" are used to help explain and clarify several fundamental ABET accreditation activities.

Categories and Subject Descriptors
K.3.2 [Computer and Information Science Education] Accreditation

General Terms: Measurement

Keywords
Accreditation, quality assurance, continual process improvement, closing the loop

1. INTRODUCTION
The premise of this paper is that using some terms that are commonly used in software engineering can help provide some guidance to those departments either undergoing an ABET accreditation visit or preparing for one. Forms and other official information about ABET and the ABET accreditation process can be found at the ABET web site. Specific criteria for accreditation of computing programs can be found in the ABET document listed in reference [1].

The activities in preparation for an ABET accreditation visit can include any or all of the following, with one likely order in which the activities occur:

- Development of measurable goals and objectives for the program which are consonant with the established mission, vision, and statements of goals and objectives for the larger program unit, usually the school or college in which the program resides, and of the college or university as a whole.
- Development of an assessment plan.
- Collection of assessment data.
- Formal invitation to ABET for a site visit.
- Sending the fees for an accreditation visit to ABET.
- Writing of the self-study report
- Creation of the binders for each course.
- Use of the assessment results to improve both individual courses and the entire curriculum.
- Evaluation of the proposed visiting team.
- Scheduling the site visit.
- Hosting the visit.
- Performing any necessary follow-up activities.
- Preparation of the final decision by ABET, based on the report of the site visit team and an assessment by ABET about the completeness and fairness of the site visit.

Many of these activities, such as the formal invitation, are self-explanatory and need not be discussed any farther. Some are time-consuming, such as the development of an assessment plan, the collection of assessment data, the writing of the self-study report, and the creation of the course binders. The purpose of these activities is also clear.

However, there often is confusion over three particular items necessary for a successful ABET accreditation visit:

- Development of an assessment plan.
- Collection of assessment data.
- Use of the assessment results to improve both individual courses and the entire curriculum.

We will concentrate on these three issues and how they can be explained to computer science faculty. The same approach can apply to the process of explaining the ABET accreditation process to faculty in information systems. The use of this terminology can be used, with perhaps somewhat less success, to faculty in the fields of information technology or computer engineering; the potential lack of clarity of explanation might occur because of unfamiliarity of faculty in these fields with the specific terminology used. The much greater unfamiliarity of typical non-computer science faculty with the vocabulary indicates that the approach suggested here will not be as successful if applied to any other engineering or technology programs that are seeking ABET accreditation.

The reader wishing to know more about some standards for systematic measurement of software products and processes is encouraged to examine references [2], [3], [4], and [5].

2. APPROACH
The approach is really quite simple – use the vocabulary of computer science as much as possible, and use the vocabulary of software engineering when that terminology is not sufficient.

We begin with the more specific vocabulary of software engineering. An "assessment plan" is a systematic way of determining how successful a program is in doing what it sets out to do. Perhaps the best way to understand the concept of an assessment plan is to use the language of software metrics. We consider both quality and process metrics.

A software quality metric describes some attribute of a software product. [6] Typical quality metrics might include the number of defects per 1000 lines of code, or the percentage of functions whose purpose is documented in the code.

A software process metric describes the efficiency and effectiveness of the process used to create the software product. [6] Typical process metrics might include the time to create each unit or module within the project, or the number of software engineers needed to complete the project

Measurement of the quality of software development processes and products is vital to the success of a software development company. Of course, many software developers, especially those new to the field, resent the time needed to create these metrics and to report them. If the company has a separate metrics team, there may be some tension between this team and developers.

Most companies of any size that develop software have some sort of quality assurance (QA) program. Often, the QA team is external to the development process. Working with the QA team is often considered even more of a burden that working with the metrics group, although QA is an essential part of the development process in many industries, especially those which develop safety-critical software.

There is another, even more general term that can be used to describe what is often the most difficult concept in program assessment – what is commonly called "continual process improvement" or "closing the loop" in ABET terminology. The goal of the ABET assessment process is for programs to use their analysis of assessment data to improve their programs and to demonstrate this use. The assessment data use can be in the form of written recommendations to provide advice to a faculty member about how to teach a course, or to a curriculum committee about how to improve a major portion of a program's curriculum. Many programs seem to have some mechanism for the recommendations of changes, but have little follow-up.

What is the most important part of continual process improvement, the part that is missing from many assessment data to actually improve the program, and to provide some evidence that the supposed "improvement" actually worked!

Perhaps the best way to explain this iterative program improvement process is to consider the vocabulary of software testing. The test-debug cycle generally consists of the following steps that are applied to programs.

1. Create the program or program unit to determine if it meets the requirements specification.
2. Use at least one test case to test the program or program unit to see if it meets the given requirements specification.
3. If the program or program unit does not meet the given requirements specification, examine the program (or program unit) and modify it until you believe it now meets the requirements specification. This process is often called debugging.
4. Test the revised program or program unit to see if, according to the test case chosen, it now meets the desired performance given in the requirements specification.
5. The first four steps in this process should be iterated over all test cases that are both considered relevant and are doable within the available resources allocated for this testing and evaluation.

Observe that step 4 is critical to make sure that the intended correction to the program or program unit actually did what it was supposed to do and that no other portion of the program or program unit was affected adversely by this presumed fix.

Let's compare the standard test-debug cycle to the ABET assessment process and the important, often misunderstood, phase of closing the loop. As part of the assessment process, the college or university's computing program's goals and objectives for its curriculum must have been created to meet the requirements that are specified in the program objectives. Individual courses should have course objectives and measurable outcomes that are consistent with the overall computing program's goals and objectives. The degree to which the course outcomes and programs are met must be addressed in the department's assessment plan.

Here are the following steps in the ABET phase that is known as closing the loop:

1. Perform the assessment of each course as indicated in the department assessment plan.
2. Use at least one assessment to test either the entire curriculum or a single course to see if it meets the given requirements specification in terms of the assessment of the student's performance.

3. If the entire curriculum or a single course does not meet the criteria for achieving the given requirements specification of mastery as shown by the assessment data, examine the entire curriculum or a single course and modify it until you believe it now meets the requirements specification. This process is often called improving the curriculum.
4. Test the revised entire curriculum or a single course to see if it now meets the desired performance given in the requirements specification, **as shown by the collection and analysis of new assessment data.** If it does not, then the attempted improvement did not work and the department should go back and revise the revision.
5. The first four steps in this process should be iterated over all test assessment cases that are both considered relevant and are doable within the available resources allocated for this testing and evaluation. Note that the assessment plan may not call for evaluation of all course materials every year during the period of overall program assessment.

Observe that step 4 is critical to make sure that the intended correction to the entire curriculum or this particular course actually did what it was supposed to do and that no other portion of the curriculum or this particular course was affected adversely by this presumed fix.

Clearly, the ABET continual process improvement, or closing the loop, is exactly the same as the test-debug cycle, with only minor changes in vocabulary.

3. CONCLUSION

Faculty understanding of the ABET accreditation process can be improved by using a vocabulary often used in the software engineering and larger computing community. These software terms, and their relation to the ABET accreditation process, are:

- Software product metrics. These are analogous to the measurement of program objectives and outcomes, and outcomes of specific courses.
- Software process metrics. These are analogous to the evaluation of how efficient the measurement process is, both in terms of coverage of the program of study and the efficiency of the coverage and evaluation.
- Software quality assurance. This is analogous to the role that the entire ABET organization, including the ABET site visit team, plays in insuring that the measurement and process improvement meets currently accepted standards in the discipline.
- The test-debug cycle. This is analogous to the cycle of making changes in response to the results of some test, then testing to see if these changes actually worked and retesting other units to see if these other unites have been affected adversely by the changes that were made.

REFERENCES

[1] ABET, "Computing Programs Criteria," www.abet.org/forms.shtml#For_Computing_Programs_Only

[2] Software Engineering Institute, The Capability Maturity Model, CMM, www.sei.cmu.edu/library/abstracts/news-at-sei/backgroundsept98pdf.cfm, Carnegie-Mellon University, Pittsburgh, Pennsylvania.

[3] Software Engineering Institute, *The Capability Maturity Model Integration CMMI* for Development, Version 1.2, www.sei.cmu.edu/library/abstracts/reports/06tr008.cfm ,Carnegie-Mellon University, Pittsburgh, Pennsylvania.

[4] Humphrey, Watts S., *Introduction to the Personal Software Process*, SEI Series in Software Engineering, Software Engineering Institute, Pittsburgh, Pennsylvania, 1997.

[5] Humphrey, Watts S., *Introduction to the Team Software Process*, SEI Series in Software Engineering, Software Engineering Institute, Pittsburgh, Pennsylvania, 2000.

[6] Moore, James, Zaman, Malia, et al., C/S2ESC/1061IEEE Standard for a Software Quality Metrics Methodology, IEEE, grouper.ieee.org/groups/index/tr_tmi.html.