Easing the Burden of Program Assessment: Web-based Tool Facilitates Measuring Student Outcomes for ABET Accreditation

Celia Schahczenski
Computer Science Department
Montana Technological University
Butte, Montana, USA
cschahczenski@mtech.edu

Michele Van Dyne
Computer Science Department
Montana Technological University
Butte, Montana, USA
mvandyne@mtech.edu

Abstract

The rapid pace of technology and social change necessitates a process of continuous program improvement for academic programs. ABET accredits educational programs, ensuring that these programs meet criteria such as continuous program improvement. Continuously collecting data, analysis of that data to determine what is, and is not, working, and updating programs accordingly consumes considerable faculty and administrative time. Software tools can help. This paper describes a tool developed and used by our department. This software tool:

1. *Reduced the burden of measuring student outcomes for members of our department for six years, and will continue to do so in the future.*
2. *Received praise by members of two ABET accreditation teams who suggested marketing the software to help other programs seeking, or maintaining, ABET accreditation.*
3. *Is undergoing enhancements for other departments in our school.*

The software was developed by students over multiple offerings of six courses in our curricula.

1. Introduction

The Computer Science Department at Montana Tech houses two undergraduate degree programs that are accredited by separate Accreditation Board for Engineering and Technology (ABET) commissions. The Computer Science (CS) degree is accredited by the Computing Accreditation Commission (CAC), and the Software Engineering (SE) degree is accredited by the Engineering Accreditation Commission (EAC). While there are differences between the accreditation standards of the two commissions, the standards for the assessment of student outcomes are similar.

Student outcomes describe what a student knows or can do by the time of graduation. Assessment is the process that identifies, collects, and prepares data to evaluate the attainment of student outcomes. Effective assessment uses relevant direct, indirect, quantitative and qualitative measures, as appropriate to the outcomes being measured. Evaluation is a process for interpreting the data and evidence accumulated through assessment processes. Evaluation determines the extent to which student outcomes are being attained. Evaluation results in decisions and actions regarding program improvement [1].

All courses in our CS and SE programs have course outcomes, that is, the knowledge and abilities we expect students to have upon successful completion of the course. Each course outcome is tied to one or more student outcomes for the program. A given course outcome, tied to a specific student outcome, contributes to a student's attainment of that student outcome for that course. Over the course of a student's progress through the curriculum, that student will encounter multiple sources of attaining a particular student outcome. However, tracking the progress of individual students is not the intention of the process – once an outcome is measured, it is no longer tied to any particular student. We maintain a matrix of which courses contribute to which student outcomes.

Each time a course is offered, the outcomes to which it is tied are measured. These are tabulated across each course every year and are presented at the annual Assessment Committee portion of the Industry Advisory Board meeting. The committee reviews the assessment results of the past year and discusses any problem areas. Because faculty tabulate and compile the results, they are generally aware of any matters that may need attention, and may come to the committee meeting with ideas for change. The committee may agree with these changes and/or suggest changes of their own.

While this approach to the assessment and evaluation of student outcomes was adequate, it was also quite burdensome for faculty members to manually

URI: https://hdl.handle.net/10125/60200
ISBN: 978-0-9981331-2-6

record, tabulate and then combine results. To facilitate the process, and to provide students with real world software experience, a student project to do exactly that was initiated in 2011. Students in the Requirements and Specifications course first defined the software requirements, Database Design students then defined its underlying database, Software Engineering students prototyped the system, Software Maintenance and Senior Design students enhanced the prototype into a working system and Verification and Validation students suggested improvements that were carried out in additional offerings of the Software Maintenance and Senior Design courses. Over the life of the project, over sixty students and eight faculty members have been involved, and the project has been used in twelve course offerings. This is described in detail by Schahczenski and Ackerman in [6]. The software was named AbOut, for ABET Outcomes.

Data was first recorded into the AbOut software in the spring semester of 2012. By the spring semester of 2013 all faculty members who teach required computing courses in either the CS or SE curricula, were required to use the software to record their assessment data. During the most recent ABET visit in 2016, members of the CAC and members of the EAC assessment teams were impressed with the software and suggested that we market the software to help other programs seeking or maintaining ABET accreditation. Faculty members who use AbOut now, and performed assessment without AbOut in the past, report an average of 50% time savings from AbOut, according to a recent small survey. Additionally, one faculty member who performed the aggregation and tabulation of results for presentation to the Assessment Committee reports an 87.5% time savings over the manual approach.

2. Related work

Many assessment tools are being used in practice. Sanders and McCartney [5] report results of two surveys distributed to the:

1. SIGCSE mailing list, and
2. Charis of all ABET-CAC accredited computer science programs

on the assessment tools they are using. Departments reported on externally produced exams, such as the Major Field Tests in Computer Science, internally produced exams, senior exit surveys, alumni surveys, employer surveys, portfolios, oral exams, and advisory board panels.

While software tools supporting the above activities abound, tools organizing and facilitating higher levels of assessment are rare and critically needed. Abunawass, Lloyd and Rudolf [4] use open-source software, COMPASS (Computer Science Program Assessment), to develop a course delivery system with portfolio analysis. Booth [3] proposes a database template for documenting program outcomes and course objectives usable by many different information technology programs, irrespective of course delivery mechanism. ACAT (ABET Course Assessment Tool), a web-based tool facilitating ABET course assessment has been prototyped, used in several trial cases and shown to be a “viable tool in data collection and reporting” [4].

Both ACAT and AbOut (the tool described in this paper) focus on ABET Criterion 3, an especially burdensome, yet insightful, part of accreditation, easily helped by automation. While ACAT is still in a prototype phase, AbOut is fully functional and has been in use for seven years. AbOut, however, was developed to support our department’s assessment process. It is only now being generalized to accommodate other departments at our school.

3. Assessment and evaluation process

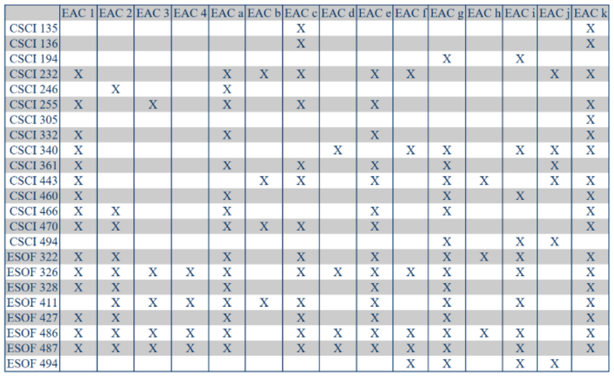
3.1 Student outcomes at the program level

Our department uses the student outcomes as stated by ABET directly. Therefore, we have one set of outcomes for the CS program, and a second set for the SE program. Each required course in the two programs contributes to a subset of the student outcomes for that program. To ensure that all outcomes are covered in at least one course, and preferably more than one, we maintain a matrix mapping student outcomes and courses. If a faculty member wishes to change course outcomes that may affect coverage of student outcomes, that change is brought to the weekly department faculty meeting for discussion before it is approved. The current matrix for the SE program is shown in Figure 1 below.

As described in the Introduction section, all of our courses have course outcomes. Those courses that are required in a given curriculum have one or more student outcomes associated with each course outcome. As an example, all students are required to take the CSCI 135 Fundamentals of Computer Science I course. Course outcome R4 states:

> R4. Students will understand and be able to use basic selection and repetition control structures. (CAC-c, i, j; EAC-k)

The parenthetical notation at the end of the course outcome indicates that this outcome contributes to specific CAC and EAC student outcomes. In this case, EAC-k is the outcome, “An ability to use the techniques, skills and modern engineering tools necessary for engineering practice”. Alone, the ability to use conditional and repetition constructs does not cover the entire student outcome, but it is one component of that outcome, and therefore contributes to it. A single course outcome may contribute to several student outcomes, and within a single course, several course outcomes may contribute to a single student outcome.



	EAC 1	EAC 2	EAC 3	EAC 4	EAC a	EAC b	EAC c	EAC d	EAC e	EAC f	EAC g	EAC h	EAC i	EAC j	EAC k
CSCI 135							X								X
CSCI 136							X								X
CSCI 194											X		X		
CSCI 232	X				X	X	X		X	X				X	X
CSCI 246		X			X										
CSCI 255	X		X		X		X		X						X
CSCI 305															X
CSCI 332	X				X				X						X
CSCI 340	X							X		X	X		X	X	X
CSCI 361	X				X		X		X		X			X	
CSCI 443	X					X	X		X		X	X		X	X
CSCI 460	X				X						X		X		X
CSCI 466	X	X			X				X		X				X
CSCI 470	X	X			X	X	X		X						X
CSCI 494											X		X	X	
ESOF 322	X	X			X		X		X		X	X	X		X
ESOF 326	X	X	X	X	X		X	X	X	X	X		X		X
ESOF 328	X	X			X				X		X				X
ESOF 411		X	X	X	X	X	X		X		X		X		X
ESOF 427	X	X			X		X		X		X				X
ESOF 486	X	X	X	X	X		X	X	X	X	X	X	X		X
ESOF 487	X	X	X	X	X		X	X	X	X	X		X		X
ESOF 494										X	X		X	X	

Figure 1. Software engineering course / student outcome matrix

3.2 Student outcomes at the course level

As described in the Introduction section, all of our courses have course outcomes. Those courses that are required in a given curriculum have one or more student outcomes associated with each course outcome. As an example, all students are required to take the CSCI 135 Fundamentals of Computer Science I course. Course outcome R4 states:

> R4. Students will understand and be able to use basic selection and repetition control structures. (CAC-c, i, j; EAC-k)

The parenthetical notation at the end of the course outcome indicates that this outcome contributes to specific CAC and EAC student outcomes. In this case, EAC-k is the outcome, “An ability to use the techniques, skills and modern engineering tools necessary for engineering practice”. Alone, the ability to use conditional and repetition constructs does not cover the entire student outcome, but it is one component of that outcome, and therefore contributes to it. A single course outcome may contribute to several student outcomes, and within a single course, several course outcomes may contribute to a single student outcome.

3.3 Course outcome measurement

All instructors teaching core courses are required to measure student attainment of course outcomes. In the previous example, the constructs of selection and repetition are taught as two separate units, and students complete a lab assignment on programming with conditionals one week, and a second one on programming with loop constructs the following week. The instructor can use the student scores on these two assignments to measure overall how well the entire class did on meeting this course outcome.

Not all course outcomes are measured by assignments. It is up to the instructor to determine how best to measure a particular course outcome. In some cases, an outcome may be measured by an exam problem or the combination of one or more exam or quiz questions. In other cases, it may be by peer evaluation, for example, in the case of participation in group projects.

3.4 Student outcome assessment at the program level

Combination of individual course outcome measurements by student outcome is done after the completion of an academic year, so all courses offered during the previous semesters (fall, spring and summer) are included. Combining the measurements is done by student outcome at this level. From the matrix in Figure 1, there are 18 courses that contribute to the student outcome EAC-k. Additionally, there may be more than one course outcome in each of those 18 courses that contributes to EAC-k.

The assessment of student outcomes is not concerned with individual student scores, but with whether, as a group, students are achieving that particular outcome. Therefore, while individual student grades will be affected by how well they perform on each of the course outcomes, the assessment process looks at how many students achieved a passing level on student outcomes. At this point of assessment, then, the percentage of students achieving a passing score (70%, or C- level) on each student outcome assessed is calculated. The percentage of students who passed is reported for the course offering. These percentages are averaged across all contributing courses in the curriculum to obtain an overall assessment of achievement of a particular student outcome.

3.5 Student outcome evaluation

So far in the process, we have described the assessment of student outcome achievement, but have not described the evaluation of that assessment. Early on in instituting this process, we defined a performance threshold of 70%. That is, the average across all contributing course offerings would meet each outcome at or above a 70% level. This was later changed to 75% as the software tool made it easy to remove failing or non-majors students from the summaries. Any outcome falling below that level is automatically addressed at the annual Assessment Committee meeting.

The Assessment Committee is responsible for reviewing the results of ongoing assessments and advising the Computer Science department on what changes might be helpful. It is comprised of members of the Industry Advisory Board, the CS Department faculty, one current upper level Computer Science /Software Engineering student, and a CS/SE alumnus who graduated in the last 4 – 5 years. If an IAB member meets the qualification for the alumnus member, then that person may fill both roles. Figure 2 provides a diagrammatic view of the process. The arrows indicate where the software automates the process.

Changes suggested by the assessment committee are added to our Improvement Log, along with a description of metrics we will use to ensure that the improvement made actually did achieve what we had hoped. Dates of implementation and measurement of the change are also documented in the log. The implementation of this entire process, then, closes the loop for continuous improvement.

4. Automation

The assessment and evaluation process just described, while effective, is tedious, error prone and burdensome when done manually. The AbOut software tool was designed and implemented to alleviate this burden.

Two levels of users may interact with the AbOut software: administrator, typically a member of the support staff; and course instructor / faculty. Valid login access is required to use the system, and this is done through the campus wide login validation system.

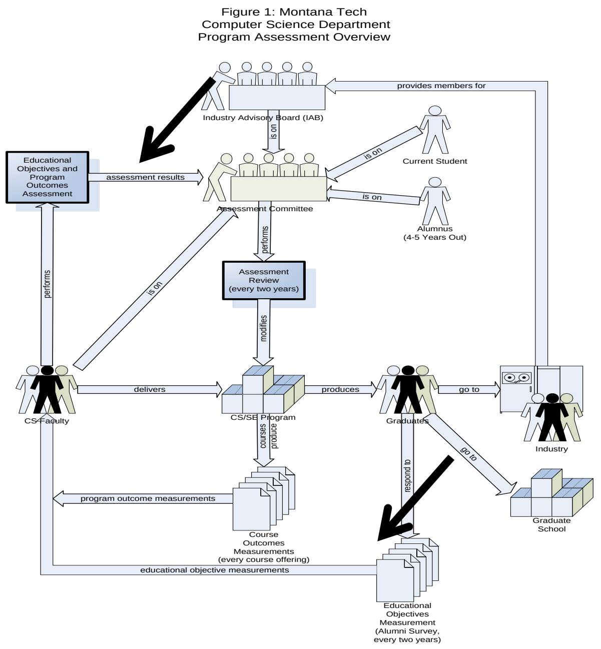


Figure 1: Montana Tech
Computer Science Department
Program Assessment Overview
Industry Advisory Board (IAB)
provides members for
is on
Educational Objectives and Program Outcomes Assessment
assessment results
Assessment Committee
is on
Current Student
is on
Alumnus (4-5 Years Out)
performs
Assessment Review (every two years)
performs
is on
modifies
CS Faculty
delivers
CS/SE Program
produces
Graduates
go to
Industry
courses produce
go to
respond to
program outcome measurements
Course Outcomes Measurements (every course offering)
Graduate School
educational objective measurements
Educational Objectives Measurement (Alumni Survey, every two years)

Figure 2. Assessment process

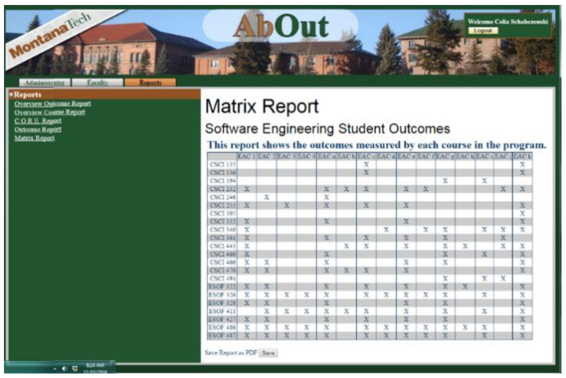
The administrator has access to create, edit and delete courses, outcomes, semesters and faculty members / users. A course may be added, deleted, associated with a set of outcomes, associated with a faculty member as instructor, and associated with an offering during a particular semester. Outcomes are generally stable, but the administrator may add, delete and edit these if necessary. Semesters can be added as needed, and the default semester can be designated. Faculty members, or those responsible for the instruction of a course may be added, deleted and set to active or inactive as necessary. Finally, the administrator can import a list of students enrolled in course offerings from the course roster.

Faculty members responsible for a given course offering may enter measurements for their course(s), may enter scores achieved by students on each of these measurements, and may add and/or delete students as necessary. Students may be removed from assessment measurement if they drop the course or do not pass it. Recall that we define an outcome as adequately achieved if 75% of students successfully completing the course achieve that outcome at a minimum of 70%. Student who do not successfully complete the course are not included in the assessment.

Finally, both the administrator and faculty users can generate reports at different levels. The reports include an Overview Outcome Report, an Overview Course Report, CORE (Course Outcome Review) Report, an Outcome Report, and a Matrix Report. These are all described in the following sections.

4.1 Student outcomes at the program level

Rather than manually maintain the course outcome / student outcome matrix that ensures we cover all student outcomes sufficiently, the AbOut software provides this report based on the association between course and outcomes as entered by the administrator. Figure 3 below shows the report generated by the AbOut software. If faculty wish to make changes to course outcomes that may affect coverage to student outcomes, the faculty refer to this table to make sure sufficient coverage is maintained. This report can be generated for the Software Engineering program, the Computer Science program, or both.



Matrix Report

Software Engineering Student Outcomes

This report shows the outcomes measured by each course in the program.

	EAC 1	EAC 2	EAC 3	EAC 4	EAC a	EAC b	EAC c	EAC d	EAC e	EAC f	EAC g	EAC h	EAC i	EAC j	EAC k
CSCI 135							X								X
CSCI 136							X								X
CSCI 194											X		X		
CSCI 232	X				X	X	X		X	X				X	X
CSCI 246		X			X										
CSCI 255	X		X		X		X		X						X
CSCI 305															X
CSCI 332	X				X				X						X
CSCI 340	X							X		X	X		X	X	X
CSCI 361	X				X		X		X		X			X	
CSCI 443	X					X	X		X		X	X		X	X
CSCI 460	X				X						X		X		X
CSCI 466	X	X			X				X		X				X
CSCI 470	X	X			X	X	X		X						X
CSCI 494											X		X	X	
ESOF 322	X	X			X		X		X		X	X			X
ESOF 326	X	X	X	X	X		X	X	X	X	X		X		X
ESOF 328	X	X			X				X		X				X
ESOF 411		X	X	X	X	X	X		X		X		X		X
ESOF 427	X	X			X		X		X		X				X
ESOF 486	X	X	X	X	X		X	X	X	X	X	X	X		X
ESOF 487	X	X	X	X	X		X	X	X	X	X		X		X

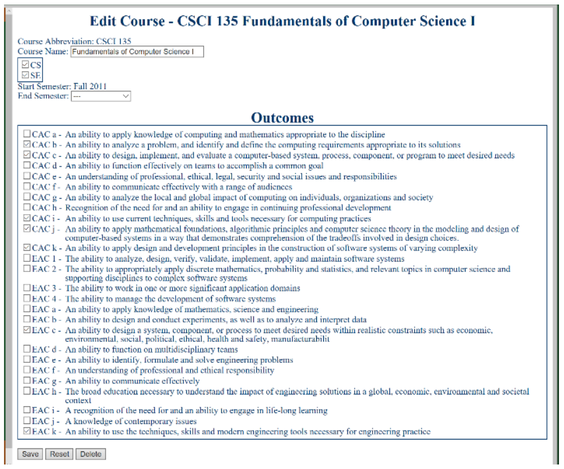
Save Report as PDF [Save]

Figure 3. SE course / student outcomes matrix generated by AbOut

4.2 Student outcomes at the course level

When a course is created or its outcomes modified, the administrator enters this information into AbOut. Figure 4 shows the page where these student outcomes may be associated with the course by the administrator.

From the faculty perspective, measurements for each of the outcomes may be added, as shown in Figure 5. The faculty member adds an assessment and chooses to which student outcome(s) the assessment applies. Only those student outcomes which are associated with that course will appear in the list. As previously described, these measurements may be an assignment, an exam problem, the combination of one or more exam questions, a quiz, or any other measure that applies to that particular course outcome.



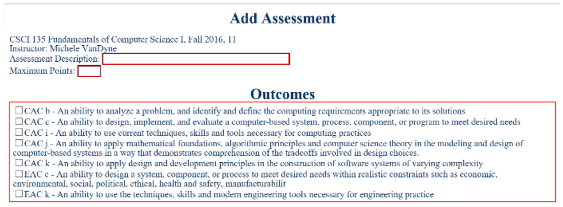
Edit Course - CSCI 135 Fundamentals of Computer Science I

Course Abbreviation: CSCI 135
Course Name: Fundamentals of Computer Science I
☑CS
☑SE
Start Semester: Fall 2011
End Semester: ---

Outcomes

- ☐CAC a - An ability to apply knowledge of computing and mathematics appropriate to the discipline
- ☑CAC b - An ability to analyze a problem, and identify and define the computing requirements appropriate to its solutions
- ☑CAC c - An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs
- ☐CAC d - An ability to function effectively on teams to accomplish a common goal
- ☐CAC e - An understanding of professional, ethical, legal, security and social issues and responsibilities
- ☐CAC f - An ability to communicate effectively with a range of audiences
- ☐CAC g - An ability to analyze the local and global impact of computing on individuals, organizations and society
- ☐CAC h - Recognition of the need for and an ability to engage in continuing professional development
- ☑CAC i - An ability to use current techniques, skills and tools necessary for computing practices
- ☑CAC j - An ability to apply mathematical foundations, algorithmic principles and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices.
- ☑CAC k - An ability to apply design and development principles in the construction of software systems of varying complexity
- ☐EAC 1 - The ability to analyze, design, verify, validate, implement, apply and maintain software systems
- ☐EAC 2 - The ability to appropriately apply discrete mathematics, probability and statistics, and relevant topics in computer science and supporting disciplines to complex software systems
- ☐EAC 3 - The ability to work in one or more significant application domains
- ☐EAC 4 - The ability to manage the development of software systems
- ☐EAC a - An ability to apply knowledge of mathematics, science and engineering
- ☐EAC b - An ability to design and conduct experiments, as well as to analyze and interpret data
- ☑EAC c - An ability to design a system, component, or process to meet desired needs within realistic constraints such as economic, environmental, social, political, ethical, health and safety, manufacturabilit
- ☐EAC d - An ability to function on multidisciplinary teams
- ☐EAC e - An ability to identify, formulate and solve engineering problems
- ☐EAC f - An understanding of professional and ethical responsibility
- ☐EAC g - An ability to communicate effectively
- ☐EAC h - The broad education necessary to understand the impact of engineering solutions in a global, economic, environmental and societal context
- ☐EAC i - A recognition of the need for and an ability to engage in life-long learning
- ☐EAC j - A knowledge of contemporary issues
- ☑EAC k - An ability to use the techniques, skills and modern engineering tools necessary for engineering practice

Figure 4. Student outcomes associated with a course: Administrator view



Add Assessment

CSCI 135 Fundamentals of Computer Science I, Fall 2016, 11
Instructor: Michele VanDyne
Assessment Description:
Maximum Points:

Outcomes

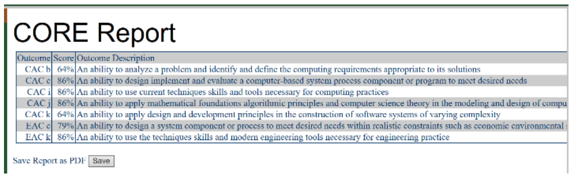
- ☐CAC b - An ability to analyze a problem, and identify and define the computing requirements appropriate to its solutions
- ☐CAC c - An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs
- ☐CAC i - An ability to use current techniques, skills and tools necessary for computing practices
- ☐CAC j - An ability to apply mathematical foundations, algorithmic principles and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices.
- ☐CAC k - An ability to apply design and development principles in the construction of software systems of varying complexity
- ☐EAC c - An ability to design a system, component, or process to meet desired needs within realistic constraints such as economic, environmental, social, political, ethical, health and safety, manufacturabilit
- ☐EAC k - An ability to use the techniques, skills and modern engineering tools necessary for engineering practice

Figure 5. Student outcomes associated with a course: Instructor addition of an assessment

4.3 Course outcome measurement

Once an assessment has been defined, the instructor can enter scores for each student enrolled in the course. The instructor continues to define and enter measurements for assessments until all student outcomes are measured. The software takes care of combining the scores when reports are generated.

A CORE Report can be generated at the course level to see how well outcomes were achieved in a particular course for a particular offering. Figure 6 shows an example of a CORE report.



CORE Report

Outcome	Score	Outcome Description
CAC b	64%	An ability to analyze a problem and identify and define the computing requirements appropriate to its solutions
CAC c	86%	An ability to design implement and evaluate a computer-based system process component or program to meet desired needs
CAC i	86%	An ability to use current techniques skills and tools necessary for computing practices
CAC j	86%	An ability to apply mathematical foundations algorithmic principles and computer science theory in the modeling and design of compu
CAC k	64%	An ability to apply design and development principles in the construction of software systems of varying complexity
EAC c	79%	An ability to design a system component or process to meet desired needs within realistic constraints such as economic environmental
EAC k	86%	An ability to use the techniques skills and modern engineering tools necessary for engineering practice

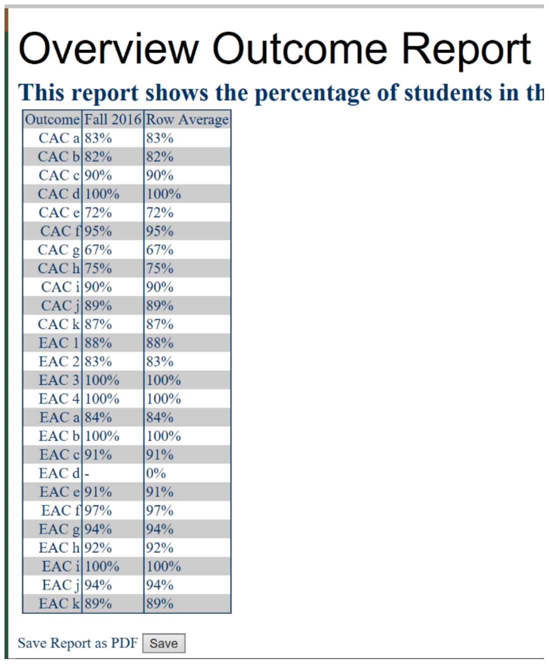
Save Report as PDF [Save]

Figure 6. CORE report showing student outcome achievement at the course level

4.4 Student outcome assessment at the program level

After all instructors have entered measurements for their courses, a report can be generated that shows the achievement of student outcomes across all courses for a given student outcome. This report can be generated by program, or for both programs, and can be run for a single semester or across multiple semesters. Typically the report is generated across all semesters in the previous academic year for use by the Assessment Committee.

Figure 7 below shows the Overview Outcome Report for all courses, both programs, for a single semester. Where a student outcome has not been covered in a single semester, the report shows a dash in the cell, as shown for outcome EAC-d in the figure.



Overview Outcome Report

This report shows the percentage of students in th

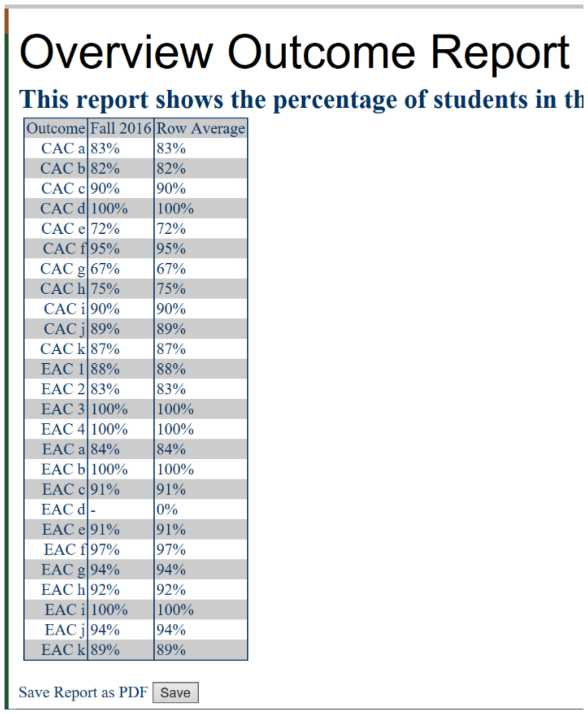
Outcome	Fall 2016	Row Average
CAC a	83%	83%
CAC b	82%	82%
CAC c	90%	90%
CAC d	100%	100%
CAC e	72%	72%
CAC f	95%	95%
CAC g	67%	67%
CAC h	75%	75%
CAC i	90%	90%
CAC j	89%	89%
CAC k	87%	87%
EAC 1	88%	88%
EAC 2	83%	83%
EAC 3	100%	100%
EAC 4	100%	100%
EAC a	84%	84%
EAC b	100%	100%
EAC c	91%	91%
EAC d	-	0%
EAC e	91%	91%
EAC f	97%	97%
EAC g	94%	94%
EAC h	92%	92%
EAC i	100%	100%
EAC j	94%	94%
EAC k	89%	89%

Save Report as PDF Save

Figure 7. Overview outcome report across one semester

4.5 Student outcome evaluation

The last step in evaluation of student outcomes is to look at those outcomes that have not met our threshold level of achievement. Looking at the data for the 2016/2017 academic year to date, these are the data that would be presented to the Assessment Committee in the fall of 2017. This data is shown in Figure 8.



Overview Outcome Report

This report shows the percentage of students in th

Outcome	Fall 2016	Row Average
CAC a	83%	83%
CAC b	82%	82%
CAC c	90%	90%
CAC d	100%	100%
CAC e	72%	72%
CAC f	95%	95%
CAC g	67%	67%
CAC h	75%	75%
CAC i	90%	90%
CAC j	89%	89%
CAC k	87%	87%
EAC 1	88%	88%
EAC 2	83%	83%
EAC 3	100%	100%
EAC 4	100%	100%
EAC a	84%	84%
EAC b	100%	100%
EAC c	91%	91%
EAC d	-	0%
EAC e	91%	91%
EAC f	97%	97%
EAC g	94%	94%
EAC h	92%	92%
EAC i	100%	100%
EAC j	94%	94%
EAC k	89%	89%

Save Report as PDF Save

Figure 8. Overview outcome report across the 2016/2017 academic year

4.6 Software development

This software was developed by students in several courses. It is a web application (HTML, PHP, JavaScript, and SQL) accessible from our departmental website. Its development served to demonstrate software engineering concepts and provide software development experience to students in the courses Requirements & Specifications, Database Design, Software Engineering, Software Maintenance, Software Verification and Validation, and Senior Design. Major challenges to the system are the navigation system, stored procedures tabulating the data, and coordinating code written by many undergraduates.

5. Results and conclusions

This project contributes to an effective continuous measurement and improvement process, supported by faculty, staff, an advisory board, guidelines, the departmental website, and the special purpose software described in this paper. The software described facilitates measuring student outcomes at a low level of granularity, and viewing results at multiple levels. The software has:

1. Reduced the burden of measuring student outcomes for members of our department for

six years, and will continue to do so in the future.
2. Received praise by members of two ABET accreditation teams who suggested marketing the software to help other programs seeking, or maintaining, ABET accreditation.
3. Is undergoing enhancements for other departments in our school.

Students benefited from envisioning, designing, developing and verifying a medium size software product. We are continuing to use this project in subsequent classes, mainly in software maintenance and senior design.

In conclusion, this paper describes a successful in-house software development project that continues to reduce the burden of low level assessment, standardizes that process, and has helped student learn about software development.

6. Future directions

6.1 Additional process automation

Entering student scores for an assessment measurement can be tedious. In some cases these scores are already kept in an Excel-spreadsheet. Enhancing the software to accept input of comma separated scores is under consideration. The AbOut software already facilitates importing and exporting a comma separated roster of student names for a course offering. The software could be adapted to accept selections from spreadsheets that were initially populated from the software.

Using the same assessment measure semester after semester within the same course would enable historic comparisons, helping faculty members see trends in student performance on those measurements. Facilitating easy copying of measurement data (assessment description, points and correspondence to student outcomes) from one semester of a course offering to another, would encourage this. Faculty members need flexibility in defining the measurements they use to collect assessment data, however, repeated use of the same measurement in different semesters can be encouraged.

6.2 Incorporating course outcomes

The current version of the software ties *student* outcomes directly to courses. The software is oblivious to *course* outcomes. Connecting assessment measurements to course outcomes, rather than student outcomes, would make the measurements align closer to course content.

Returning to the earlier example, a course outcome for CSCI 135 Fundamentals of Computer Science is R4:

> R4. Students will understand and be able to use basic selection and repetition control structures. (CAC-c, i, j; EAC-k)

This *course* outcome maps to the *student* outcomes CAC-c, i, j and EAC-k. Assessment measurements associated with this outcome are translatable by the software to the corresponding student outcomes.

6.3 Document consistency

In the future, the software can automatically generate website reports. Currently administrators generate reports via the AbOut tool and add those reports to the website. Inevitably, AbOut data is changed but we forget to update the departmental website. The system can automatically propagate changes to the website, or propagate the changes when requested by an administrator.

If AbOut maintains course outcomes, this information could also be exported directly to the departmental website. Additionally, it could be exported in a format easily added to course syllabi.

6.4 External availability

Currently access to the software is limited to two roles: administrator and faculty. Defining a third observer role would enable members of an accreditation team to view, but not write, all FERPA-acceptable data. While individual measurement data is associated with students enrolled in courses, the remaining data is associated with students so is not sensitive, and can be publicly available.

Part of the accreditation evaluation process includes keeping copies of graded assignments. Expanding the system to store images of graded measurement items would provide one central, easily accessible repository of assessment data and its source.

6.5 Enhancing for other programs

Montana Tech is primarily a STEM institution, and most, if not all, departments in the School of Engineering undergo ABET accreditation. Each department has autonomy in choosing how to address assessment and evaluation criteria specified by ABET. Extending AbOut to allow different approaches and different measurement methods would make the software more portable within our institution. Both the Electrical Engineering and the Petroleum Engineering departments have expressed interest in using AbOut in their assessment process. A senior design project this year began the process of enhancing AbOut to accommodate other departments. This effort is called Stout, for Student Outcomes.

Finally, during our past ABET accreditation visit, ABET evaluators recommended that the department look into marketing our assessment software to other institutions having or seeking ABET accreditation. It is expected that this would require significant exploration of the needs of other departments and enhancements to the software. We would like to encourage any external institutions interested in using the software to contact either of the authors.

7. Acknowledgments

We would like to thank the entire faculty in the Computer Science Department for contributing to the development, use, and critique of the AbOut software. Dr. Frank Ackerman has been extremely instrumental in developing the requirements for both the current system and for extending the system to other departments. We would also like to thank all the students who, through several years of coursework, are responsible for the actual development and deployment of the system.

10. References

[1] ABET Criterion 3, http://www.abet.org/accreditation/accreditation-criteria/accreditation-alerts/changes-to-the-eac-accreditation-criteria-general-criterion-3-student-outcomes/, ABET, accessed May 21, 2018.

[2] Abunawass A. Lloyd W. Rudolph E. "COMPASS: a CS program assessment project", Proceedings of the 9th Annual SIGCSE Coference on Innovation and Technoloyg in Comptuer Sicnece Education, 127-131, 2004.

[3] Booth, L. "A database to promote continuous program imporvement", Proceedings of the 7th conference on Information Technology Education, 83-88, 2006.

[4] Essa, E. Dittrich, A. Dascalu, S. Harris F. "ACAT: A Web-based Software Tool To Facilitate Course Assessment for ABET Accredication", 7th International Conference on Information Technology, 88-93, 2010.

[5] Sanders, K. McCartney, R. "Program Assessment Tools in Computer Science: A Report from the Trenches", Proceedings of the 34th SIGCSE Technical Symposium on Computer Science Education, 31-35, 2003.

[6] Schahczenski, C. Ackerman, F. "A Successful Multi-Course Project", Journal of Computer Sciences in Colleges, 33(1), 202-208, October 2017.