Oregon Institute of Technology
Computer Systems Engineering Technology Department
Software Engineering Technology Program Assessment Plan
2014-2015

I. Introduction

The Software Engineering Technology (SET) program was implemented in Klamath Falls in 1984 and was initially accredited by TAC of ABET in 1991. The Portland program was established in Fall 1996 under the same accreditation and is currently located on the Wilsonville campus. The Associate degree was accredited by TAC of ABET in 2009. The program has continuously evolved as industrial changes have warranted.

Note: The Institutional Research office was unable to provide current data for the following tables due to personnel change-over. As a results, the data from the previous year is included in this year's report.

A. Enrollment

Table 1.1 shows the number of students that have listed Software Engineering Technology (SET) as their major at the end of Week 4, Fall Term 2013.

Table 1.1 SET Enrollment Data Fall 2013

Campus	Frosh.	Soph.	Junior	Senior	Masters	PostBac	NonAdmit-UG	NonAdmit-G	Total
Klamath	33	27	25	39	0	2	1	0	**127**
Wilsonville	8	10	21	37	0	16	3	0	**95**
Totals	**41**	**37**	**46**	**76**	**0**	**18**	**4**	**0**	**222**

Table 1.2 shows the number of students that have designated that they are pursuing a concurrent degree with the Computer Engineering Technology (CET) program as their major at the end of Week 4, Fall Term 2013.

Table 1.2 Concurrent SET and CET Enrollment Data Fall 2013

Campus	Frosh.	Soph.	Junior	Senior	Masters	PostBac	NonAdmit-UG	NonAdmit-G	Total
Klamath	2	11	6	13	0	0	0	0	**32**
Wilsonville	na	na	na	na	na	na	na	na	**na**
Totals	**2**	**11**	**6**	**13**	**0**	**0**	**0**	**0**	**32**

B. Retention

The following retention data in Table 1.3 shows the percentage of students that returned to the program for their second year. This is data is only for the Klamath Falls campus.

Table 1.3 Klamath Retention Data

	Returning		
	N	Y	Total
2009	10	63	73
	13.70%	86.30%	100.00%
2010	20	54	74
	27.03%	72.97%	100.00%
2011	13	61	74
	17.57%	82.43%	100.00%
2012			

C. Employment Data

The data shown in Table 1.4 shows the data collected on the student graduate survey. This information is for the Bachelor degree only.

Table 1.4 Bachelor Degree Employment Data

Campus	Year	Number of Respondents	Full-time Employed	Employment Not Reported	Average Salary	Maximum Salary
Klamath	2014	24	21	3	65.16	100,000.00

The data shown in Table 1.5 shows the data collected on the student graduate survey. This information is for the Associate degree.

Table 1.5 Associate Degree Employment Data

Campus	Year	Number of Respondents	Full-time Employed	Employment Not Reported	Average Salary	Maximum Salary
Klamath	2010	0	0	0	NA	NA

II. Mission, Objectives and Student Learning Outcomes

On February 19, 2015, the software faculty met with our Industry Advisory Board and reviewed and approved its program mission, objectives and student learning outcomes. It was agreed that we would adopt ABET's learning outcomes as our Student Learning Outcomes instead of maintaining a separate list and have to show the correlation between the two lists.

The mission statement, objectives and program outcomes for the baccalaureate program are located on the OIT website at www.oit.edu/provost/learningoutcomes/cset/swbs. The associate program's mission statement, objectives and program outcomes are located at www.oit.edu/provost/learningoutcomes/cset/swae.

Bachelor Program Mission

The mission of the Software Engineering Technology (SET) Bachelor's Degree program within the Computer Systems Engineering Technology (CSET) Department at Oregon Institute of Technology is to prepare our students for productive careers in industry and government by providing an excellent education incorporating industry-relevant, applied laboratory based instruction in both the theory and application of software engineering. The program is to serve a constituency consisting of our alumni, our employers, and our Industrial Advisory Board. Major components of the SET program's mission in the CSET Department are:

I. To educate a new generation of Software Engineering Technology students to meet current and future industrial challenges and emerging software trends.

II. To promote a sense of scholarship, leadership, and professional service among our graduates.

III. To enable our students to create, develop, apply, and disseminate knowledge within the software development environment.

IV. To expose our students to cross-disciplinary educational programs.

V. To provide government and high tech industry employers with graduates in software engineering and related professions.

Bachelor Program Educational Objectives

The Program Educational Objectives of OIT's Software Engineering Technology program are to produce graduates that:

A. Use their knowledge of engineering to creatively and innovatively solve difficult computer systems problems.

B. Regularly engage in exploring, learning and applying state-of-the-art hardware and software technologies to the solution of computer systems problems.

C. Will be an effective software development team member that contributes innovative software design solutions to the resolution of business, scientific or government computer systems problems.

D. Will communicate effectively and successfully, both individually and within multi-disciplinary teams.

Bachelor Program Student Learning Outcomes

Software Engineering Technology baccalaureate graduates will have demonstrated:

a. an ability to select and apply the knowledge, techniques, skills, and modern tools of the discipline to broadly-defined engineering technology activities;

b. an ability to select and apply a knowledge of mathematics, science, engineering, and technology to engineering technology problems that require the application of principles and applied procedures or methodologies;

c. an ability to conduct standard tests and measurements; to conduct, analyze, and interpret experiments; and to apply experimental results to improve processes;

d. an ability to design systems, components, or processes for broadly-defined engineering technology problems appropriate to program educational objectives;

e. an ability to function effectively as a member or leader on a technical team;

f. an ability to identify, analyze, and solve broadly-defined engineering technology problems;

g. an ability to apply written, oral, and graphical communication in both technical and non-technical environments; and an ability to identify and use appropriate technical literature;

h. an understanding of the need for and an ability to engage in self-directed continuing professional development;

i. an understanding of and a commitment to address professional and ethical responsibilities including a respect for diversity;

j. a knowledge of the impact of engineering technology solutions in a societal and global context; and

k. a commitment to quality, timeliness, and continuous improvement.

Associate Program Mission

The mission of the Software Engineering Technology (SET) Associate Degree program within the Computer Systems Engineering Technology (CSET) Department at Oregon Institute of Technology is to prepare our students for entry level careers in the software industry and government by providing applied laboratory based instruction. The program is to serve a constituency consisting of our alumni, our employers, and our Industrial Advisory Board. Major components of the SET program's mission in the CSET Department are:

I. To provide a new generation of Software Engineering Technology students with a solid background in computer programming.

II. To enable our students to create, develop and apply knowledge within a technical software environment.

III. To provide government and high tech industry employers with entry level graduates in computer programming and related professions.

Associate Program Educational Objectives

The Program Educational Objectives of OIT's Software Engineering Technology program are to produce graduates that:

A. Assist in solving computer systems problems using their knowledge of computer programming.

B. Regularly engage in learning and applying state-of-the-art hardware and software technologies to the solution of computer systems problems

C. Will communicate effectively and successfully in the workplace.

Associate Program Outcomes

Software Engineering Technology associates graduates will have demonstrated:

a. an ability to apply the knowledge, techniques, skills, and modern tools of the discipline to narrowly defined engineering technology activities;

b. an ability to apply a knowledge of mathematics, science, engineering, and technology to engineering technology problems that require limited application of principles but extensive practical knowledge;

c. an ability to conduct standard tests and measurements, and to conduct, analyze, and interpret experiments;

d. an ability to function effectively as a member of a technical team;

e. an ability to identify, analyze, and solve narrowly defined engineering technology problems;

f. an ability to apply written, oral, and graphical communication in both technical and non-technical environments; and an ability to identify and use appropriate technical literature;

g. an understanding of the need for and an ability to engage in self-directed continuing professional development;

h. an understanding of and a commitment to address professional and ethical responsibilities, including a respect for diversity; and

i. a commitment to quality, timeliness, and continuous improvement.

III. Three-Year Cycle for Assessment of Student Learning Outcomes

The department assesses the program educational objectives and student learning outcomes on a three-year cycle. During the six-year ABET cycle, the objectives and learning outcomes will thus be fully assessed twice.

All appropriate accreditation documents are housed on a SharePoint site maintained by the department. All department members have access to this site, but the documents are not viewable by the general public. The public can view the baccalaureate outcomes at www.oit.edu/provost/learningoutcomes/cset/swbs and the associate outcomes at www.oit.edu/provost/learningoutcomes/cset/swbs.

Bachelor Degree Assessment Cycle

We changed Student Learning Outcomes mid-year. Since we started the year with the old outcomes, we finished this year with those same outcomes. Beginning next year, we will assess the new outcomes. Table 3-1 shows the old plan. Table 3-2 shows the plan with the new outcomes.

Table 3-1: Baccalaureate Outcome Assessment Timeline

#	**Learning Outcomes**	**12-13**	**13-14**	**14-15**	**15-16**	**16-17**	**17-18**
1	an ability to identify, formulate, and solve software engineering problems, including the specification, design, implementation, and testing of software systems that meet specification, performance, maintenance and quality requirements for a major software project	**X**					
2	the ability to elicit, analyze and specify software requirements through a productive working relationship with various stakeholders of the project			**X**	**X**		**X**
3	an understanding of the core areas of software engineering			**X**	**X**		**X**
4	an ability to function effectively on teams	**X(I)**					
5	an understanding of professional, ethical and social responsibility	**X(I)**					
6	a recognition of the need for, and an ability to engage in life-long learning		**X**			**X(I)**	
7	knowledge of and ability to apply discrete math, probability and statistics			**X**	**X**		**X(I)**
8	an ability to convey technical material through oral presentation and interaction with an audience		**X**			**X(I)**	
9	an ability to convey technical material through written reports which satisfy accepted standards for writing style		**X**			**X(I)**	
10	an ability to evaluate the impact of potential solutions to software engineering problems in a global society, using their knowledge of contemporary issues and emerging software engineering trends, models, tools, and techniques		**X**				

Note: (I) represents an ISLO to be assessed.

Table 3-2 Assessment plan for the new Student Learning Outcomes

#	Learning Outcome	2015-2016	2016-2017	2017-2018
a	an ability to select and apply the knowledge, techniques, skills, and modern tools of the discipline to broadly-defined engineering technology activities	X		
b	an ability to select and apply a knowledge of mathematics, science, engineering, and technology to engineering technology problems that require the application of principles and applied procedures or methodologies		X	
c	an ability to conduct standard tests and measurements; to conduct, analyze, and interpret experiments; and to apply experimental results to improve processes			X
d	an ability to design systems, components, or processes for broadly-defined engineering technology problems appropriate to program educational objectives	X		
e	an ability to function effectively as a member or leader on a technical team	X		
f	an ability to identify, analyze, and solve broadly-defined engineering technology problems			X
g	an ability to apply written, oral, and graphical communication in both technical and non-technical environments; and an ability to identify and use appropriate technical literature		X(I)	
h	an understanding of the need for and an ability to engage in self-directed continuing professional development		X(I)	
i	an understanding of and a commitment to address professional and ethical responsibilities including a respect for diversity	X(I)		
j	a knowledge of the impact of engineering technology solutions in a societal and global context			X
k	a commitment to quality, timeliness, and continuous improvement		X(I)	

Associate Degree Assessment Cycle

Table 3-3: Associate Outcome Assessment Timeline

#	**Learning Outcomes**	**12-13**	**13-14**	**14-15**	**15-16**	**16-17**
1	an ability to identify, formulate, and solve computer programming problems, including the specification, design, implementation, and testing of programs that meet specification, performance, maintenance and quality requirements	**X**			**X**	
2	an understanding of the core areas of software engineering			**X**		
3	an understanding of professional, ethical and social responsibility	**X(I)**			**X(I)**	
4	a recognition of the need for, and an ability to engage in life-long learning		**X**			**X**
5	an ability to communicate through oral presentation and interaction with an audience		**X**			**X**
6	an ability to convey technical material through written reports which satisfy accepted standards for writing style		**X**			**X**

Note: (I) represents an ISLO

Table 3-4 Assessment plan for the new Student Learning Outcomes

#	**Learning Outcome**	**2015-2016**	**2016-2017**	**2017-2018**
a	an ability to apply the knowledge, techniques, skills, and modern tools of the discipline to narrowly defined engineering technology activities	X		
b	an ability to apply a knowledge of mathematics, science, engineering, and technology to engineering technology problems that require limited application of principles but extensive practical knowledge		X	
c	an ability to conduct standard tests and measurements, and to conduct, analyze, and interpret experiments			X
d	an ability to function effectively as a member of a technical team	X		
e	an ability to identify, analyze, and solve narrowly defined engineering technology problems			X
f	an ability to apply written, oral, and graphical communication in both technical and non-technical environments; and an ability to identify and use appropriate technical literature		X(I)	
g	an understanding of the need for and an ability to engage in self-directed continuing professional development		X(I)	
h	an understanding of and a commitment to address professional and ethical responsibilities, including a respect for diversity	X(I)		
i	a commitment to quality, timeliness, and continuous improvement		X(I)	

IV. Summary of Assessment Activities

From the three years cycle matrix, the 2014-2015 outcomes are extracted, courses/instructors are chosen and specific assignments are given to assess the outcomes. Table 4.1 and 4.2 below outline the assignments for 2014-2015 for respectively Klamath Falls and Wilsonville campuses.

Bachelor Degree				
Learning Outcome	Comparable ABET A-K	Direct#1	Direct#2	Indirect
BS#2: ability to elicit, analyze and specify software requirements through a productive working relationship with various stakeholders of the project	C: an ability to conduct standard tests and measurements; to conduct, analyze, and interpret experiments; and to apply experimental results to improve processes	Course- CST415 Instructor-Bishop Quarter- Fall Assignment-Paper **COMPLETED F'14**	Course- cst334 Instructor-caldwell Quarter- Spring Assignment-Proposal	Exit Survey **Completed Fall '14**
BS#3: understanding of the core areas of software engineering (data structures…operating systems…)	F: an ability to identify, analyze, and solve broadly-defined engineering technology problems	Course- cst240 Instructor-Scevers Quarter- Winter Assignment-???	Course- cst352 Instructor-Howard Quarter- spring Assignment-Exam **Completed S'15**	Exit Survey **Completed Fall '14**
BS#7 knowledge of and ability to apply discrete math, probability and statistics	an ability to select and apply a knowledge of mathematics, science, engineering and technology to engineering technology problems that require the application of principles and applied procedures or methodologies	Course- cst162 Instructor-Nguyen Quarter- Fall Assignment- Quiz **COMPLETED F'14**	Course- cst466 Instructor-Nguyen Quarter- Spring Assignment-Quiz **Completed S'15**	Exit Survey **Completed Fall '14**
Associate Degree				
Learning Outcome	Comparable ABET A-K	Direct#1	Direct#2	Indirect
AE#2: understanding of the core areas of software engineering	E: an ability to identify, analyze, and solve narrowly defined engineering technology problems	Course- cst240 Instructor-Scevers Quarter- Winter Assignment-???		Exit survey **Completed Fall '14**

Table 4.1 Klamath Falls Campus Assessment Assignments for 2014-2015

	Bachelor Degree			
#	Learning Outcome	Direct#1	Direct#2	Indirect
	BS#2: ability to elicit, analyze and specify software requirements through a productive working relationship with various stakeholders of the project	CST 334 – Project Proposal Sp 2015. Assignment: Project Proposal paper	CST 432 – Senior Project Development Sp 2015. Assignment: Project test plan	Klamath Falls Campus Exit Survey Sufficient
	BS#3: an understanding of the core areas of software engineering. (data structures, theory of computation, operating systems, compilers, programming languages, computer architecture).	CST 432 Sp 2015. Assignment: Project Report	CST 352 Sp 2015 Assignment: Lab	Klamath Falls Campus Exit Survey Sufficient
	BS# 7- knowledge of and ability to apply discrete math, probability and statistics	CST 130 – Computer Organization W 2015 Assignment: Test	CST 229 – Intro to Grammars F 2014 Assignment: Homework	Klamath Falls Campus Exit Survey Sufficient

Table 4.2 OIT Wilsonville Campus Assessment Assignments for 2014-2015

ASSESSMENT RESULTS
TABLE 4.1 and 4.2

BS#2: ability to elicit, analyze and specify software requirements through a productive working relationship with various stakeholders of the project
Assessment Activity #1 (BS 2)- Klamath

Course used for assessment: CST 415 – Computer Networks
Instructor/Evaluator: David Bishop (Klamath)
Student level: Senior
Term of administration: Fall 2014
Assessed work: Requirement Analysis for Networked Application
Type of assessment: Direct

Data Collection Date: 11/9/14 Coordinator: David Bishop

Assessment Method: Task was to perform a requirements analysis for a networked application which performs according to the customer requirements. The deliverables to be included as a result of this analysis were: Functional Requirements, UML Class Diagrams, Use Cases, Dataflow Diagrams, & Sequence Diagrams.

Performance Criteria	Measurement Scale	Minimum Acceptable Performance	Results
Functional Requirements	1-4 according to rubric	3.0	3.39
UML Class Diagrams	"	"	3.33
Use Cases	"	"	3.43
Dataflow Diagrams	"	"	3.5
Sequence Diagrams	"	"	3.3

Evaluation 11/10/14 (date)

The performance passed standard.

Actions 11/10/14 (date)

No formal action is required.

Assessment Activity #2 (BS2)- Klamath

Course used for assessment: CST 334 – Project Proposal
Instructor/Evaluator: Calvin Caldwell (Klamath)
Student level: Junior
Term of administration: Spring 2014
Number of students: 24
Assessed work: Proposal
Type of assessment: Direct

Students wrote a proposal for their senior project. The proposals were evaluated based on organization, scope, requirements, and risk.

	Some Proficiency	**Proficiency**	**High Proficiency**
Organization	**16%**	**0%**	**84%**
Scope	**16%**	**0%**	**84%**
Requirements	**16%**	**0%**	**84%**
Risks	**16%**	**0%**	**84%**

Most of the students in the class demonstrated high proficiency. In the last assessment cycle (2011-2012) the department was short-staffed and was thus unable to complete this assessment. As a result, we aren't able to compare these results to the last cycle. However, the level of proficiency suggests that no action is required.

Action: No action required.

Assessment Activity #3 (BS 2)

Method used for assessment:	Exit Survey
Instructor/Evaluator:	Nguyen
Student level:	Senior Graduates
Term of administration:	Graduating Class 2014
Number of students:	19
Assessed work:	Survey
Type of assessment:	Indirect

To assess this outcome for the institution, graduating students of 2014 were asked to complete an exit survey, the result which pertains to BS 2 is shown below

AGREE SURVEY

#	Description	Strongly Disagree	Disagree	Agree	Strongly Agree	Total Responses	Mean
2	ability to elicit, analyze and specify software requirements through a productive working relationship with various stakeholders of the project	0	1	14	4	19	3.16

Assessment Activity #1 (BS2)- Wilsonville

Course used for assessment: CST 334 – Project Proposal
Instructor/Evaluator: Jay Bockelman (Wilsonville)
Student level: Junior
Term of administration: Spring 2015
Number of students: 22
Assessed work: Project Proposal
Type of assessment: Direct
Data Collection Date: 6/8/2015 Coordinator: Jay Bockelman

Assessment Method: Task was to write a project proposal for the student's Senior Project. A template was supplied that specified the sections with examples of the details required. Customers/stakeholders of this project had to be identified, along with the success criteria and the risks involved in delivering this product to the customers.

Performance Criteria	Measurement Scale	Minimum Acceptable Performance	Results
Functional Requirements	1-4 according to rubric	3.0	3.3
Identify Customers	"	"	3.5
Project Success Criteria	"	"	3.6
Project Risks	"	"	3.8
Summer Research	"	"	3.3

Evaluation 6/8/2015 (date)

The performance passed standard.

Actions 6/8/2015 (date)

No formal action is required.

Assessment Activity #2 (BS2)- Wilsonville

Course used for assessment: CST 432 – Senior Project Development
Instructor/Evaluator: Jay Bockelman (Wilsonville)
Student level: Senior
Term of administration: Spring 2015
Number of students: 19
Assessed work: Project Test Plan
Type of assessment: Direct

Assessment Method: Task was to write a project test plan for the Senior Project. A template was supplied that specified the sections with examples of the details required. Required were the following sections:

- Unit Test Plan
- Integration Test Plan
- Performance Test Plan
- User Acceptance Test Plan
- Test Summary

Performance Criteria	Measurement Scale	Minimum Acceptable Performance	Results
Unit Test Plan	1-4 according to rubric	3.0	3.3
Integration Test Plan	"	"	3.2
Performance Test Plan	"	"	3.6
User Acceptance Test Plan	"	"	3.8
Test Summary	"	"	3.3

Evaluation 6/8/2015 (date)

The performance passed standard.

Actions 6/8/2015 (date)

Action Required:
Though the performance passed standard, it was evident that students did not fully integrate a test methodology into their Senior Project planning and activities. Some students indicated performing testing after development was completed.
Some students actively used a bug tracking system, but not all did.
Suggestions are to place more emphasis on test driven development, and analyzing test results, perhaps in the CST236 Software Testing class..

BS#3: an understanding of the core areas of software engineering. (data structures, theory of computation, operating systems, compilers, programming languages, computer architecture).

Assessment Activity #1(BS 3/AE#2)- Klamath

Course used for assessment: CST 240 – UNIX
Instructor/Evaluator: Troy Scevers
Student level: Sophomore
Term of administration: Winter 2015
Number of students: 23
Assessed work: Problem #5 from final exam
Type of assessment: Direct

Assessment Date:
Data Collection Date: 3/17/14 Coordinator: Troy Scevers

Assessment Method: Students (23 SET Students) in CST 240 were given a question on the final to solve a n! (factorial) algorithm using the python scripting language.

Performance Criteria	Measurement Scale	Minimum Acceptable Performance	Results
Understanding the programming problem	1 -4 according to rubric	3	18 Of 23 78%
Plan to solve the problem	1 -4 according to rubric	3	18 Of 23 78%
Solution (Python Code)	1 -4 according to rubric	2	21 Of 23 91%

Evaluation:
As expected most students were able to understand the basic algorithm. They had trouble writing actual working code on the final. Most students made very simple syntax errors that would have been quickly fixed if writing on an actual system. Overall I was pleased with the outcome of this problem and feel the sophomores express a good understanding of the core areas of software implementation. They need more practice at implementation which will come in the junior and senior year.

Action Required: None

Assessment Activity #2 (BS 3)- Klamath

Spring 2014

Course used for assessment:	CST 352 – Operating Systems
Instructor/Evaluator:	Phil Howard
Student level:	Junior
Term of administration:	Spring 2015
Number of students:	24
Assessed work:	Exam questions
Type of assessment:	Direct

Assessment method: Two questions from an exam were tailored to assess the student learning outcome. Although the topics for the questions come from Operating Systems, the questions are really looking for an understanding of the core areas of software engineering. These questions were as follows:

> 1) In Labs 6 and 7, it was necessary to keep track of threads that were not ready to run. Given the following requirements, identify what data structure you would use to keep track of this information assuming development time was not an issue. Explain why you chose that data structure. Note: I'm looking for a good choice of data structure and a good justification for that choice.
>
> a) Must be able to keep track of an arbitrary number of threads
> b) Must be able to quickly find a thread control block given a thread_id
>
> 2) You have two systems. The only difference between the two is that one is running S5FS, the other is running FFS. Both systems have 10,000 RPM drives. You replace both drives with 15,000 RPM drives. How would you expect the performance of each system to change? Justify your answer.

The following rubric was used to assess these questions:

Question	1	2	3	4
Data structure for thread list	Did not show that they understood the implications of the question.	Understood the question, but did not make a good choice of data structure and did not justify their choice.	Either made a non-optimal choice on data structure or failed to adequately justify their choice.	Made a good choice of data structure and provided a good justification.
Performance change with faster drive	Did not show that they understood the implications of the question.	Showed minimal understanding of the underlying file system layouts, but understood the performance implications of disk RPM.	Explained how disk RPM affects file system performance, but did not explain the difference between FFS and S5FS.	Explanation of effects of disk RPM on both S5FS and FFS.

Evaluation of data

Criteria	Measurement Scale	Minimal Acceptable Performance	Average Results	Number of students above/below standard
Data structure for thread list	1-4 based on rubric	3.0	3.4	19/6
Performance change with faster drive	1-4 based on rubric	3.0	2.75	12/12

The students were able to select a good data structure for a particular need and were able to justify their selection. However, students were less able to analyze the performance related impacts of a disk based data structure.

Actions: Next year's class

Need to spend more time analyzing the time implications of various hardware/data structure combinations.

Assessment Activity #3 (BS 3/AE 2)

Method used for assessment: Exit Survey
Instructor/Evaluator: Nguyen
Student level: Senior Graduates
Term of administration: 2014 Graduating Class
Number of students: 19
Assessed work: Survey
Type of assessment: Indirect

To assess this outcome for the institution, graduating students of 2014 were asked to complete an exit survey, the result which pertains to BS#3 is shown below

AGREE SURVEY

#	Description	Strongly Disagree	Disagree	Agree	Strongly Agree	Total Responses	Mean
3	**C**ore Area	0	0	7	12	19	3.63

Assessment Activity #1 (BS 3) - Wilsonville

Course used for assessment: CST 432
Instructor/Evaluator: Jay Bockelman
Student level: Senior
Term of administration: Spring 2015
Number of students: 20
Assessed work: Project Report
Type of assessment: Direct
Data Collection Date: 6/10/15 Coordinator: Jay Bockelman
Assessment Method: Students (20 total) in CST 432 were required to write a comprehensive report describing their senior project, their test methodologies, results of those tests, and a discussion of the results. A template report was provided with a description of expected content in each section.

Performance Criteria	Measurement Scale	Minimum Acceptable Performance	Results
Project description	1-4 according to rubric	3.3	3.8
Development approach	"	"	4
Testing approach	"	"	3.8
Test results	"	"	3.6
Analysis of test results	"	"	3.8

Evaluation 6/10/2015 (date)
The performance passed standard.
Actions 6/10/2016 (date)
No formal action is required.

Assessment Activity #2 (BS 3) - Wilsonville

Course used for assessment: CST 352
Instructor/Evaluator: Jay Bockelman
Student level: Junior
Term of administration: Spring 2015
Number of students: 16
Assessed work: Lab
Type of assessment: Direct
Data Collection Date: 5/16/15 Coordinator: Jay Bockelman

Assessment Method: Students (16 total) in CST 352 were a Lab assignment that tested student's ability to analyze, design and implement a multi-threaded prime number calculator using the 'C' programming language on a Linux system.

Performance Criteria	Measurement Scale	Minimum Acceptable Performance	Results
Identify and analyze the problem requirements	1-4 according to rubric	3.3	3.8
Design a multi-threaded solution	"	"	3.5
Implement and test solution	"	"	3.8
Validate results	"	"	3.6

Evaluation 5/16/2015 (date)
The performance passed standard.
Actions 5/16/2015 (date)
No formal action is required.

BS 7- knowledge of and ability to apply discrete math, probability and statistics

Assessment Activity #1 (BS 7)- Klamath

Course used for assessment: CST 162 – Introduction to Digital Logic
Instructor/Evaluator: Phong Nguyen
Student level: Freshman
Term of administration: Fall 2014
Number of students: 32
Assessed work: Quiz
Type of assessment: Direct

Assessment Method:
Data Collection Date: 11/05/14 Coordinator: Phong Nguyen

Assessment Method: Students (32 total) in CST 162 were given a set of specifications to a digital logic K-Map problem. They are next required to follow a discrete math method to solve the K-Map and test the result.

Performance Criteria	Measurement Scale	Minimum Acceptable Performance	Results
Understanding Specifications Do Truth Table	1-4 according to rubric	3.3	3.812
Plan to Solve Boolean Algebra	"	"	4
Carry out Plan K-Map	"	"	3.531
Evaluating Check solution to Truth Table	"	"	3.656
Solution Logic Diagram	"	"	3.843

3.8125 4 3.53125 3.65625 3.84375

Evaluation 11/15/14 (date)

The performance passed standard.

Actions 11/15/14 (date)

No formal action is required.

Assessment Activity #2 (BS 7)- Klamath

Course used for assessment: CST 466 – Embedded Security
Instructor/Evaluator: Phong Nguyen
Student level: Senior
Term of administration: Spring 2015
Number of students: 13
Assessed work: Quiz
Type of assessment: Direct

Data Collection Date: 5/13/2015 Coordinator: Phong Nguyen

Assessment Method: Quiz

The students in CST 466 were given a set of math questions in an exam involving the encryption and decryption calculations for an RSA set of keys

Performance Criteria	Measurement Scale	Minimum Acceptable Performance	Results
Understanding Diffie-Hellman	1-4 according to rubric	3.0	3.0
Understanding RSA	1-4 according to rubric	3.0	2.92

Evaluation: 5/13/15

The students met the standard in Diffie-Hellman, but not in RSA. The majority of the RSA shortcoming had to do with the Digital Signature portion of the RSA

Actions: Next year's class

Need to add an actual example with numbers in lecture. Will do this by assigning a project doing RSA digital signatures next year.

Assessment Activity #3 (BS 7)

Method used for assessment: Exit Survey
Instructor/Evaluator: Phong Nguyen
Student level: Senior Graduates
Term of administration: 2014 Graduates
Number of students: 19
Assessed work: Survey
Type of assessment: Indirect

PROFICIENCY SURVEY

# Question	No/Limited Proficiency	Some Proficiency	Proficiency	High Proficiency	Total Responses	Mean
7 Math	0.00%	10.53%	63.16%	26.32%	19	3.16

AGREE SURVEY

#	Description	Strongly Disagree	Disagree	Agree	Strongly Agree	Total Responses	Mean
7	Discrete math	1	3	14	1	19	2.79

Assessment Activity #1 (BS 7)- Wilsonville

Course used for assessment: CST 130 – Computer Organization
Instructor/Evaluator: Jay Bockelman
Student level: Freshman
Term of administration: Winter 2015
Number of students: 20
Assessed work: Test
Type of assessment: Direct

Data Collection Date: 3/16/15 Coordinator: Jay Bockelman

Assessment Method: Students (20 total) in CST 130 were given a test that covered conversion and representation of number systems, CPU organization, Addressing modes, memory models, and assembly language.

Performance Criteria	Measurement Scale	Minimum Acceptable Performance	Results
Understanding machine organization (#1,2,3,9)	1-4 according to rubric	3.3	3.8
Understanding memory organization(#4,10)	"	"	3.6
Understanding number representations (#5,6)	"	"	3.8
Understanding assembly language (#7,8,13)	"	"	3.6
Understanding CPU architecture (#11,12,14)	"	"	3.8

Evaluation 3/16/2015 (date)
The performance passed standard.
Actions 3/16/2016 (date)
No formal action is required.

Assessment Activity #2 (BS 7)- Wilsonville

Course used for assessment: CST 229 – Intro to Grammars
Instructor/Evaluator: Sherry Yang
Student level: Juniors
Term of administration: Fall 2014
Number of students: 13
Assessed work: HomeWork assignment
Type of assessment: Direct

Data Collection Date: 10/05/14 Coordinator: Sherry Yang

Assessment Method: Students (13 total) in CST 229 were given a homework assignment consisting of 9 questions on Set Theory and Discrete Math. Sample question:

A class contains 9 girls and 3 boys.

a) In how many ways can the teacher choose a committee of four students?
b) How many of them will contain at least 1 boy.

Performance Criteria	Measurement Scale	Minimum Acceptable Performance	Results
Understanding Basic SET theory operations (6 questions)	1-4 according to rubric	3.3	3.7
Ability to apply discrete math principles (3 questions)	"	"	3.6

Evaluation 10/5/14 (date)
The performance passed standard.
Actions 10/5/14 (date)
No formal action is required.

V. Summary of Student Learning Outcomes Results

A. BS 2 - ability to elicit, analyze and specify software requirements through a productive working relationship with various stakeholders of the project

PREVIOUS RESULTS:

Not assessed in 2011-12

CURRENT RESULTS:

CST-415 Computer Networks **Prof. David Bishop** **Fall 2014**

Strengths: Standard was met
Weaknesses: None.
Action Items: None

CST-334 Project Proposal **Prof.** Calvin Caldwell **Winter 2015**

Strengths: Standard was met
Weaknesses: None.
Action Items: None

CST-334 Project Proposal **Prof.** Jay Bockelman **Spring 2015**

Strengths: Standard was met
Weaknesses: None.
Action Items: None

CST-432 Senior Project Development **Prof.** Jay Bockelman **Spring 2015**

Strengths: Standard was met
Weaknesses: None.
Action Items: Place more emphasis on test driven development and analyzing test results in CST236 Software Testing class.

CLOSING THE LOOP FROM 2011-2012 RESULTS

1. Weakness: BS#2 not assessed in 2011-12

Action taken: Did assess in next cycle, 2014-15.

2. Weakness: In 2011-2012, only one direct assessment was done in any assessment.

Action taken: Two direct assessments were conducted.

4. No indirect assessment was done in 2011-2012
Action taken: an Indirect Assessment was introduced in this cycle

1) BS 3 - an understanding of the core areas of software engineering. (data structures, theory of computation, operating systems, compilers, programming languages, computer architecture).

PREVIOUS RESULTS:

Not assessed in 2011-2012

CURRENT RESULTS:

CST-240 Unix **Prof.** Troy Scevers **Winter 2015**

Strengths: Standard was met
Weaknesses: None.
Action Items: None

CST-352 Operating Systems **Prof.** Philip Howard **Spring 2015**

Strengths: Standard was partially met
Weaknesses: Students did not do an adequate job in analyzing the performance implications of an on-disk data structure.
Action Items: Spend more time discussing performance issues in hardware/software systems.

CST-432 Senior Project Development **Prof.** Jay Bockelman **Spring 2015**

Strengths: Standard was met
Weaknesses: None.
Action Items: None

CST-352 Operating Systems **Prof.** Jay Bockelman **Fall 2014**

Strengths: Standard was met
Weaknesses: None.
Action Items None.

Closing the Loop

This outcome was not assessed in the previous cycle. In this cycle we did two direct assessments at each campus as well as an indirect assessment. The assessments show that our students are doing well, and we now have solid data to compare with the next assessment cycle.

1) BS 7 - knowledge of and ability to apply discrete math, probability and statistics

PREVIOUS RESULTS:

MATH 327 Discrete Mathematics
Strengths: None
Weaknesses: Done in MATH 327 instead of CST classes. Also a paper was used instead of actually solving a problem
Action Items: Redo next cycle in a CST class

CURRENT RESULTS:

CST-162 Introduction to Digital Logic **Prof.** Phong Nguyen **Fall 2014**

Strengths: Standard was met
Weaknesses: None.
Action Items: None

CST 466 – Embedded Security **Prof.** Phong Nguyen **Spring 2015**

Strengths: Standard was met
Weaknesses: Student's understanding of Digital Signature portion of the RSA was not as strong as it should be.
Action Items: Need to add an actual example with numbers in lecture. Will do this by assigning a project doing RSA digital signatures next year.

CST-131 Computer Organization **Prof.** Jay Bockelman **Winter 2015**

Strengths: Standard was met
Weaknesses: None.
Action Items: None

CST-229 Intro to Grammars **Prof.** Sherry Yang **Fall 2014**

Strengths: Standard was met
Weaknesses: None.
Action Items None.

CLOSING THE LOOP

1. The faculty assessed this outcome in MATH 327 Discrete Mathematics spring term 2012. The assessment consisted of a paper exploring various discrete mathematical theories scored by a rubric.
2. Indirect assessment was done this cycle by way of an Exit Interview Survey.

Appendix A
Course Mapping Matrices

(Note: Courses shaded in red will be used to assess the respective SLOs)

BS#6, AE#4 - a recognition of the need for, and an ability to engage in life-long learning				
Course	**Teach**	**Eval**		
CST 102 – Introduction to Computer Systems	L	L		**E**- Extensive - a major focus of the course
CST 105 – Introduction to Computer Systems III	L	L		
CST 116 – C++ Programming I				**M**- Moderate - subject explicitly discussed in and class materials provided
CST 126 – C++ Programming II				**L**- Little explicit discussion - student may gain the skill due to activities
CST 130 – Computer Organization				
CST 131 – Computer Architecture				
CST 136 – Object Oriented Programming with C++				
CST 162 – Introduction to Digital Logic				
CST 211 – Data Structures		L		
CST 223 - Concepts of Programming Languages				
CST 229 – Grammars				
CST 236 - Software System Testing				
CST 238 – GUI Programming		M		
CST 240 – Unix				
CST 250 – Assembly Language Programming				
CST 276 - Software Design Patterns				
CST 316 – Software Process Management		E		
CST 326 – Software Design and Implementation I		E		
CST 336 – Software Design and Implementation II		E		
CST 320 – Compiler Methods				
CST 324 – Database Systems and Design		L		
CST 334 – Project Proposal				
CST 352 – Operating Systems		M		
CST 412 – Senior Development Project		E		
CST 422 – Senior Development Project		E		
CST 432- Senior Development Project		E		
CST 415 – Computer Networks		E		

AE#4 - a recognition of the need for, and an ability to engage in life-long learning				
Course	**Teach**	**Eval**		
CST 102 – Introduction to Computer Systems	L	L		**E**- Extensive - a major focus of the course
CST 105 – Introduction to Computer Systems III	L	L		
CST 116 – C++ Programming I				**M**- Moderate - subject explicitly discussed in and class materials provided
CST 126 – C++ Programming II				**L**- Little explicit discussion - student may gain the skill due to activities
CST 130 – Computer Organization				
CST 131 – Computer Architecture				
CST 136 – Object Oriented Programming with C++				
CST 162 – Introduction to Digital Logic				
CST 211 – Data Structures		L		
CST 223 - Concepts of Programming Languages				
CST 236 - Software System Testing				
CST 238 – GUI Programming		M		
CST 240 – Unix				
CST 276 - Software Design Patterns				

BS#8 - an ability to convey technical material through oral presentation and interaction with an audience				
Course	**Teach**	**Eval**		
CST 102 – Introduction to Computer Systems				**E**- Extensive - a major focus of the course
CST 105 – Introduction to Computer Systems III	L	L		
CST 116 – C++ Programming I				**M**- Moderate - subject explicitly discussed in and class materials provided
CST 126 – C++ Programming II				**L**- Little explicit discussion - student may gain the skill due to activities
CST 130 – Computer Organization				
CST 131 – Computer Architecture				
CST 136 – Object Oriented Programming with C++				
CST 162 – Introduction to Digital Logic				
CST 211 – Data Structures				
CST 223 - Concepts of Programming Languages				
CST 229 – Grammars				
CST 236 - Software System Testing				
CST 238 – GUI Programming	L	L		
CST 240 – Unix				
CST 250 – Assembly Language Programming				
CST 276 - Software Design Patterns				
CST 316 – Software Process Management		L		
CST 326 – Software Design and Implementation I		L		
CST 336 – Software Design and Implementation II		L		
CST 320 – Compiler Methods				
CST 324 – Database Systems and Design				
CST 334 – Project Proposal				
CST 352 – Operating Systems				
CST 412 – Senior Development Project				
CST 422 – Senior Development Project				
CST 432 – Senior Development Project	L	L		
CST 415 – Computer Networks				

AE#5 depending on canceling #5 - an ability to convey technical material through oral presentation and interaction with an audience				
Course	**Teach**	**Eval**		
CST 102 – Introduction to Computer Systems	L	L		**E**- Extensive - a major focus of the course
CST 105 – Introduction to Computer Systems III	L	L		
CST 116 – C++ Programming I				**M**- Moderate - subject explicitly discussed in and class materials provided
CST 126 – C++ Programming II				**L**- Little explicit discussion - student may gain the skill due to activities
CST 130 – Computer Organization				
CST 131 – Computer Architecture				
CST 136 – Object Oriented Programming with C++				
CST 162 – Introduction to Digital Logic				
CST 211 – Data Structures		L		
CST 223 - Concepts of Programming Languages				
CST 236 - Software System Testing				
CST 238 – GUI Programming		M		
CST 240 – Unix				
CST 276 - Software Design Patterns				

BS#9 - an ability to convey technical material through written reports which satisfy accepted standards for writing style				
Course	**Teach**	**Eval**		
CST 102 – Introduction to Computer Systems	L	L		**E**- Extensive - a major focus of the course
CST 105 – Introduction to Computer Systems III	L	L		
CST 116 – C++ Programming I				**M**- Moderate - subject explicitly discussed in and class materials provided
CST 126 – C++ Programming II				**L**- Little explicit discussion - student may gain the skill due to activities
CST 130 – Computer Organization				
CST 131 – Computer Architecture				
CST 136 – Object Oriented Programming with C++				
CST 162 – Introduction to Digital Logic				
CST 211 – Data Structures				
CST 223 - Concepts of Programming Languages				
CST 229 – Grammars				
CST 236 - Software System Testing				
CST 238 – GUI Programming				
CST 240 – Unix				
CST 250 – Assembly Language Programming				
CST 276 - Software Design Patterns				
CST 316 – Software Process Management		M		
CST 326 – Software Design and Implementation I		M		
CST 336 – Software Design and Implementation II		M		
CST 320 – Compiler Methods				
CST 324 – Database Systems and Design				
CST 334 – Project Proposal	L	E		
CST 352 – Operating Systems				
CST 412 – Senior Development Project				
CST 422 – Senior Development Project				
CST 432 – Senior Development Project				
CST 415 – Computer Networks				

AE#6 - an ability to convey technical material through written reports which satisfy accepted standards for writing style				
Course	**Teach**	**Eval**		
CST 102 – Introduction to Computer Systems	L	L		**E**- Extensive - a major focus of the course
CST 105 – Introduction to Computer Systems III	L	L		
CST 116 – C++ Programming I				**M**- Moderate - subject explicitly discussed in and class materials provided
CST 126 – C++ Programming II				**L**- Little explicit discussion - student may gain the skill due to activities
CST 130 – Computer Organization				
CST 131 – Computer Architecture				
CST 136 – Object Oriented Programming with C++				
CST 162 – Introduction to Digital Logic				
CST 211 – Data Structures				
CST 223 - Concepts of Programming Languages				
CST 236 - Software System Testing				
CST 238 – GUI Programming				
CST 240 – Unix				
CST 276 - Software Design Patterns				

BS#8 - an ability to convey technical material through oral presentation and interaction with an audience				
Course	**Teach**	**Eval**		
CST 102 – Introduction to Computer Systems				**E**- Extensive - a major focus of the course
CST 105 – Introduction to Computer Systems III				
CST 116 – C++ Programming I				**M**- Moderate - subject explicitly discussed in and class materials provided
CST 126 – C++ Programming II				**L**- Little explicit discussion - student may gain the skill due to activities
CST 130 – Computer Organization				
CST 131 – Computer Architecture				
CST 136 – Object Oriented Programming with C++				
CST 162 – Introduction to Digital Logic				
CST 211 – Data Structures				
CST 223 - Concepts of Programming Languages				
CST 229 – Grammars				
CST 236 - Software System Testing				
CST 238 – GUI Programming				
CST 240 – Unix				
CST 250 – Assembly Language Programming				
CST 276 - Software Design Patterns				
CST 316 – Software Process Management				
CST 326 – Software Design and Implementation I				
CST 336 – Software Design and Implementation II				
CST 320 – Compiler Methods				
CST 324 – Database Systems and Design				
CST 334 – Project Proposal				
CST 352 – Operating Systems				
CST 412 – Senior Development Project				
CST 422 – Senior Development Project				
CST 432 – Senior Development Project				
CST 415 – Computer Networks				
ANTH 452 - Globalization	E	E		