Annual Assessment Report of the CCSU Computer Science Department covering the 2013 – 2014 Academic Year
Contents
Preamble ....................................................................................................................................................... 8
Section 1. Learning Outcomes...................................................................................................................... 9
Section 2. Findings...................................................................................................................................... 13
Student Outcome (a): An ability to apply knowledge of computing and mathematics appropriate to the discipline. 13
Student Outcome (b): An ability to analyze a problem, and identify the computing requirements appropriate to its solution. .......................................................... 16
Student Outcome (c): An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs .......................................................... 18
Student Outcome (d): An ability to function effectively on teams to accomplish a common goal .................................................. 20
Student Outcome (e): An understanding of professional, ethical, and social responsibilities ........ 21
Student Outcome (f): An ability to communicate effectively ....................................................................................... 22
Student Outcome (g): An ability to analyze the impact of computing on individuals, organizations, and society, including ethical, legal, security, and global policy issues .......................................................... 24
Student Outcome (h): Recognition of the need for and an ability to engage in continuing professional development ............................................................................................................................................................. 25
Student Outcome (i): An ability to use current techniques, skills, and tools necessary for computing practice .................................................. 26
Student Outcome (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 .................................................. 27
Student Outcome (k): An ability to apply design and development principles in the construction of software systems of varying complexity ............................................................................................................................................................. 29
Section 3. Analysis..................................................................................................................................... 30
SO (a): An ability to apply knowledge of computing and mathematics appropriate to the discipline .. 30
SO (b): An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution ............................................................................................................................................................. 30
SO (c): An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs ............................................................................................................................................................. 30
SO (d): An ability to function effectively on teams to accomplish a common goal .................................................. 30
SO (e): An understanding of professional, ethical, and social responsibilities .................................................. 30
SO (f): An ability to communicate effectively ............................................................................................................................................................. 30
SO (g): An ability to analyze the impact of computing on individuals, organizations, and society, including ethical, legal, security, and global policy issues ............................................................................................................................................................. 30
SO (h): Recognition of the need for and an ability to engage in continuing professional development ............................................................................................................................................................. 30
SO (i): An ability to use current techniques, skills, and tools necessary for computing practice ........ 30
SO (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 ............................................................................................................................................................. 30
SO (k): An ability to apply design and development principles in the construction of software systems of varying complexity ................................................................. 31

Section 4. Use of Results ............................................................................................................................ 32

SO (a): An ability to apply knowledge of computing and mathematics appropriate to the discipline .. 32

SO (b): An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution ..................................................................................................................... 32

SO (c): An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs ...................................................................................................... 32

SO (d): An ability to function effectively on teams to accomplish a common goal ................................ 32

SO (e): An understanding of professional, ethical, and social responsibilities ..................................... 32

SO (f): An ability to communicate effectively ............................................................................................ 32

SO (g): An ability to analyze the impact of computing on individuals, organizations, and society, including ethical, legal, security, and global policy issues ......................................................................................... 32

SO (h): Recognition of the need for and an ability to engage in continuing professional development ................................................................................................................................. 32

SO (i): An ability to use current techniques, skills, and tools necessary for computing practice ....... 32

SO (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 ................................................................................................................................. 33

SO (k): An ability to apply design and development principles in the construction of software systems of varying complexity ............................................................................................................................ 33

Section 5. General Education .................................................................................................................... 34

General Education Summary ............................................................................................................... 34

General Education Findings ................................................................................................................ 35

Section 6. Assessment Plan ....................................................................................................................... 38

Appendix – Additional Material ................................................................................................................... 39

CCSU Computer Science Department Honors Program: Curriculum and Continuous Quality Improvement Process, Metrics, and Definitions .................................................................................................................. 39

1. Industry Advisory Board ....................................................................................................................... 39
2. Department Meetings .......................................................................................................................... 40
3. Program Educational Objectives ......................................................................................................... 41
4. Student Outcomes ................................................................................................................................... 42
5. Assessment .......................................................................................................................................... 43

5.1. Program-level Assessment ........................................................................................................... 43

5.2. Assessment of Student Outcomes .................................................................................................. 45

5.2.1 SO a: An ability to apply knowledge of computing and mathematics appropriate to the discipline ................................................................. 46

5.2.2 SO b: An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution ................................................................................................................ 47

5.2.3 SO c: An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs ................................................................................... 48

5.2.4 SO d: An ability to function effectively on teams to accomplish a common goal .................... 49

5.2.5 SO e: An understanding of professional, ethical, and social responsibilities ....................... 50
5.2.6  SO f: An ability to communicate effectively ................................................................. 51

5.2.7  SO g: An ability to analyze the impact of computing on individuals, organizations, and society, including ethical, legal, security, and global policy issues ................................................................. 52

5.2.8  SO h: Recognition of the need for and an ability to engage in continuing professional development .......................................................................................................................... 53

5.2.9  SO i: An ability to use current techniques, skills, and tools necessary for computing practice ........................................................................................................................................... 54

5.2.10  SO 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 ............................................................................................................................ 55

5.2.11  SO k: An ability to apply design and development principles in the construction of software systems of varying complexity ................................................................................................................ 57

5.3.  Course-level Assessment ........................................................................................................... 58

6.  Evaluation ......................................................................................................................................... 59

6.1.  Attainment of Student Outcomes ................................................................................................. 59

6.2.  Course-level Evaluation ................................................................................................................. 59

6.3.  Course Adjustments as a Result of Evaluation ............................................................................. 59

7.  References ........................................................................................................................................ 60

Document change log ................................................................................................................................. 60
**Program Summary**

Department: Computer Science  
Report Preparer: Stan Kurkovsky  
Program Name and Level: BS in Computer Science

<table>
<thead>
<tr>
<th>Program Assessment Question</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) <strong>URL:</strong> Provide the URL where the learning outcomes (LO) can be viewed.</td>
<td><a href="http://www.cs.ccsu.edu/mission.html">http://www.cs.ccsu.edu/mission.html</a></td>
</tr>
<tr>
<td>2) <strong>Assessment Instruments:</strong> For each LO, what is the source of the data/evidence, other than GPA, that is used to assess the stated outcomes? (e.g., capstone course, portfolio review and scoring rubric, licensure examination, etc.)</td>
<td>Varies. See attached CQI Process document.</td>
</tr>
<tr>
<td>3) <strong>Interpretation:</strong> Who interprets the evidence? (e.g., faculty, Admn. assistant, etc.). If this differs by LO, provide information by LO.</td>
<td>Individual faculty for each Course Learning Outcome, entire department for Student Outcomes</td>
</tr>
<tr>
<td>4) <strong>Results:</strong> Since the most recent full report, state the conclusion(s) drawn, what evidence or supporting data led to the conclusion(s), and what changes have been made as a result of the conclusion(s).</td>
<td>As a result from ABET/CAC feedback, the entire program assessment process has been completely redesigned.</td>
</tr>
<tr>
<td>5) <strong>Strengths:</strong> What about your assessment process is working well?</td>
<td>Remains to be determined by the next ABET/CAC accreditation review in 2010.</td>
</tr>
<tr>
<td>6) <strong>Improvements:</strong> What about your assessment process needs to improve? (a brief summary of changes to assessment plan should be reported here)</td>
<td>The CQI Process has been adopted as of Spring 2014 semester and will be completely rolled out by the end of the 2014/15 academic year. It will not be known until then if the assessment process needs any improvement.</td>
</tr>
</tbody>
</table>
## General Education Summary

**Department: Computer Science**

**General Education LO Assessed:** To strengthen quantitative skills

**Report Preparer:** Stan Kurkovsky

<table>
<thead>
<tr>
<th>General Education Question</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) <strong>Courses:</strong> General Education course(s) taught</td>
<td>CS 110, CS 113</td>
</tr>
<tr>
<td>2) <strong>Assessment Instruments:</strong> What data/evidence, other than GPA, are used to assess the stated CCSU General Education outcomes? (e.g., capstone course, portfolio review, licensure examination, etc.)</td>
<td>Questions embedded in midterm and final exams.</td>
</tr>
<tr>
<td>3) <strong>Interpretation:</strong> Who interprets the evidence? (e.g., faculty, Admn. assistant, etc.) If this differs by XX course, provide information by XX course.</td>
<td>Faculty.</td>
</tr>
<tr>
<td>4) <strong>Results:</strong> Since the most recent full report, state the conclusion(s) drawn, what evidence or supporting data led to the conclusion(s), and what changes have been made as a result of the conclusion(s).</td>
<td>Majority of students perform at or above the Acceptable level.</td>
</tr>
<tr>
<td>5) <strong>Strengths:</strong> What about your assessment process is working well?</td>
<td>Assessment process is working as designed.</td>
</tr>
<tr>
<td>6) <strong>Improvements:</strong> What about your assessment process needs to improve? (changes to assessment plan should be reported here)</td>
<td>It will most likely need to be revised if/when the new general education structure is adopted.</td>
</tr>
</tbody>
</table>
**General Education Summary:**

**Department:** Computer Science

**General Education LO Assessed:** To develop information fluency and computer literacy

**Report Preparer:** Stan Kurkovsky

<table>
<thead>
<tr>
<th>General Education Question</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) <strong>Courses:</strong> General Education course(s) taught</td>
<td>CS 110, CS 113</td>
</tr>
<tr>
<td>2) <strong>Assessment Instruments:</strong> What data/evidence, other than GPA, are used assess the stated CCSU General Education outcomes? (e.g., capstone course, portfolio review, licensure examination, etc.)</td>
<td>Questions embedded in midterm and final exams.</td>
</tr>
<tr>
<td>3) <strong>Interpretation:</strong> Who interprets the evidence? (e.g., faculty, Admn. assistant, etc.). If this differs by XX course, provide information by XX course.</td>
<td>Faculty.</td>
</tr>
<tr>
<td>4) <strong>Results:</strong> Since the most recent full report, state the conclusion(s) drawn, what evidence or supporting data led to the conclusion(s), and what changes have been made as a result of the conclusion(s).</td>
<td>Majority of students perform at or above the Acceptable level.</td>
</tr>
<tr>
<td>5) <strong>Strengths:</strong> What about your assessment process is working well?</td>
<td>Assessment process is working as designed.</td>
</tr>
<tr>
<td>6) <strong>Improvements:</strong> What about your assessment process needs to improve? (changes to assessment plan should be reported here)</td>
<td>It will most likely need to be revised if/when the new general education structure is adopted.</td>
</tr>
</tbody>
</table>
Preamble

The Computer Science Department prepares students for graduate study and for positions in computer-related professions, including computer systems analysis, software development, Internet programming, security, software engineering, and database administration.

The curriculum conforms to the accreditation standards of ABET, the recognized accreditor for college and university programs in computer science and engineering. The degree programs cover the full range of computer sciences and include a foundation in mathematics and science. This background is expected for professional positions and for graduate study in computer science.

The Department offers two degrees:

**B. S. Computer Science (Honors)** – this degree is accredited by the Computing Accreditation Commission of ABET (CAC of ABET), recognized by the Council on Higher Education Accreditation for computer science accreditation.

**B. S. Computer Science (Alternative)** – this degree is intended for transfer students and for students who change their major and so do not have the time for all the mathematics and science that the Honors degree requires.

The same computer science courses are used in both programs. Since the programs are similar, assessment data covers both degrees. Most of the assessment instruments are administered to all students together, and partitioning the results into two groups would be difficult. Students frequently shift between the two degrees. The learning outcomes (educational objectives and student outcomes) are the same for both programs.

Most significant changes.

- Based on the feedback from ABET/CAC and from the CS Department Industrial Advisory Board, Program Educational Objectives have been updated:
  - Objective 1. Graduates will apply a broad understanding of the fundamental theories, concepts, and applications of computer science in their career.
  - Objective 2. Graduates will be engaged in a wide range of careers in computer science and information technology.
  - Objective 3. Graduates will communicate effectively, both orally and in writing.
  - Objective 4. Graduates will engage in life-long learning or continue with graduate study in computer science.
  - Objective 5. Graduates will act ethically and with social responsibility in their careers.
- Based on the feedback from ABET/CAC, the department has adopted a Curriculum and Continuous Quality Improvement Process (CQI Process), which guides all assessment activities (program- and course-level) for the CS-Honors program (attached). CQI Process outlines all assessment instruments and measurable performance criteria for all Student Outcomes and their correspondence to individual courses. The vast majority of the assessment instruments newly adopted in the CQI Process are incompatible with those used prior to 2013/14 Academic Year.
- CS Honors degree has been significantly revised:
  - A capstone sequence is now required;
  - The structure of program electives has been simplified.
Section 1. Learning Outcomes

In agreement with ABET’s definitions, Program Educational Objectives describe what graduates are expected to attain within a few years after graduation. Program educational objectives are based on the needs of the program’s constituencies. Objectives are the long-term goals of the degree program. Our objectives are published at http://cs.ccsu.edu/mission.html

The following objectives were discussed and approved by the department and our Industrial Advisory Board.

- Objective 1. Graduates will apply a broad understanding of the fundamental theories, concepts, and applications of computer science in their career.
- Objective 2. Graduates will be engaged in a wide range of careers in computer science and information technology.
- Objective 3. Graduates will communicate effectively, both orally and in writing.
- Objective 4. Graduates will engage in life-long learning or continue with graduate study in computer science.
- Objective 5. Graduates will act ethically and with social responsibility in their careers.

Student Outcomes describe what students are expected to know and be able to do by the time of graduation, similar to the “Learning Outcomes” of CCSU documents. These relate to the knowledge, skills, and behaviors that students acquire as they progress through the program. An outcome is often supported by a learning outcome of a particular course. Following are the student outcomes expected by Criteria for Accrediting Computing Programs (Computer Accreditation Commission, October 29, 2011), www.abet.org.
The program enables students to attain, by the time of graduation:

**Student Outcome (a):** An ability to apply knowledge of computing and mathematics appropriate to the discipline.

**Student Outcome (b):** An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution.

**Student Outcome (c):** An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs.

**Student Outcome (d):** An ability to function effectively on teams to accomplish a common goal.

**Student Outcome (e):** An understanding of professional, ethical, legal, security and social issues and responsibilities.

**Student Outcome (f):** An ability to communicate effectively with a range of audiences.

**Student Outcome (g):** An ability to analyze the impact of computing on individuals, organizations, and society.

**Student Outcome (h):** Recognition of the need for and an ability to engage in continuing professional development.

**Student Outcome (i):** An ability to use current techniques, skills, and tools necessary for computing practice.

**Student Outcome (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.

**Student Outcome (k):** An ability to apply design and development principles in the construction of software systems of varying complexity.

Program educational objectives are supported by the Student Outcomes as indicated in the table below:

<table>
<thead>
<tr>
<th>Program Educational Objectives</th>
<th>Student outcomes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>a</td>
</tr>
<tr>
<td>PEO 1 – fundamentals</td>
<td>x</td>
</tr>
<tr>
<td>PEO 2 – careers</td>
<td></td>
</tr>
<tr>
<td>PEO 3 – communication</td>
<td>x</td>
</tr>
<tr>
<td>PEO 4 – life-long learning</td>
<td></td>
</tr>
<tr>
<td>PEO 5 – ethics</td>
<td>x</td>
</tr>
</tbody>
</table>
Student outcomes are supported by the courses included in both Computer Science degrees as detailed in the table below. Major Field Test (MFT) is also used for Student Outcome assessment.

<table>
<thead>
<tr>
<th>Curricular components and faculty coordinators</th>
<th>Student outcomes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>a</td>
</tr>
<tr>
<td>Core courses and requirements</td>
<td></td>
</tr>
<tr>
<td>CS 151 Zlatesareva</td>
<td>x</td>
</tr>
<tr>
<td>CS 152 Pevac</td>
<td>A</td>
</tr>
<tr>
<td>CS 153 Kjell</td>
<td>x</td>
</tr>
<tr>
<td>CS 253 Zlatesareva</td>
<td>x</td>
</tr>
<tr>
<td>CS 254 Kjell</td>
<td>A</td>
</tr>
<tr>
<td>CS 354 Markov</td>
<td>x</td>
</tr>
<tr>
<td>CS 355 Kurkovsky</td>
<td>x</td>
</tr>
<tr>
<td>CS 385 Markov</td>
<td>x</td>
</tr>
<tr>
<td>CS 410 Kurkovsky</td>
<td>x</td>
</tr>
<tr>
<td>CS 498 Kurkovsky</td>
<td>x</td>
</tr>
<tr>
<td>MFT</td>
<td>A</td>
</tr>
<tr>
<td>Advanced electives</td>
<td></td>
</tr>
<tr>
<td>CS 407**</td>
<td></td>
</tr>
<tr>
<td>CS 415 Kurkovsky</td>
<td>x</td>
</tr>
<tr>
<td>CS 423 Kjell</td>
<td>x</td>
</tr>
<tr>
<td>CS 425*</td>
<td>x</td>
</tr>
<tr>
<td>CS 460 Pevac</td>
<td>x</td>
</tr>
<tr>
<td>CS 462 Zlatesareva</td>
<td>x</td>
</tr>
<tr>
<td>CS 463 Pevac</td>
<td>x</td>
</tr>
<tr>
<td>CS 464 Abdollahzadeh</td>
<td>x</td>
</tr>
<tr>
<td>CS 465*</td>
<td>x</td>
</tr>
<tr>
<td>CS 473*</td>
<td>x</td>
</tr>
<tr>
<td>CS 481 Abdollahzadeh</td>
<td>x</td>
</tr>
<tr>
<td>CS 483*</td>
<td>x</td>
</tr>
<tr>
<td>CS 490 Williams</td>
<td>x</td>
</tr>
<tr>
<td>CS 491*</td>
<td>x</td>
</tr>
<tr>
<td>CS 492 Williams</td>
<td>x</td>
</tr>
<tr>
<td>CS 495</td>
<td>x</td>
</tr>
<tr>
<td>Auxiliary</td>
<td></td>
</tr>
<tr>
<td>CS 290**</td>
<td>x</td>
</tr>
<tr>
<td>CS 300*</td>
<td>x</td>
</tr>
<tr>
<td>CS 301*</td>
<td>x</td>
</tr>
<tr>
<td>CS 398*</td>
<td>x</td>
</tr>
<tr>
<td>CS 499*</td>
<td>x</td>
</tr>
</tbody>
</table>

- x – a curricular component supports the corresponding Student Outcome;
- A – a curricular component is used for assessment of the corresponding Student Outcome;
• * – a rarely offered course;
• ** – a topics course that may support a variety of SO’s, depending on the offered topic.
Section 2. Findings

Student Outcome (a): An ability to apply knowledge of computing and mathematics appropriate to the discipline.

Student Outcome (a) is assessed in CS 152, 254.

Embedded assessment in final exams in CS 152 – Computer Science II
Assessment Instrument: CS 152 covers lists including arrays of objects and ArrayLists. In addition, it includes fundamental object oriented concepts such as inheritance, polymorphism via inheritance, and polymorphism via interfaces. Students learn interfaces, abstract classes, input and output files, protected visibility modifier, the use of super reference in methods and in constructor, and exceptions. Parameter passing knowledge is extended by including formal parameters of interface type, and abstract class type while actual parameters can be objects from the class that implements the interface or objects from descendants of such classes. Additional topics covered in the course include linear and binary searching, and selection and insertion sorting, recursion, basic GUI components and containers, and event driven programming. Starting in the Spring of 2014 linked lists were also covered in CS152. The students are required to implement five to six projects, and to take two tests and final exam.

Measurable Performance Criteria: The final exam is comprehensive and reflects what students have learned, after implementing five to six projects, and after taking two tests during the semester. The average score of the final exams is converted from 0 to 100 point scale into 1 to 4 point scale. That score is used to assess the Student Outcome a) (SO-a) as follows: Not Met (1.00-1.99), Minimally Attained (2:00-2.79), Met (2.80-3.59), Exceeded (3.60-4.00).

<table>
<thead>
<tr>
<th>Academic year</th>
<th>2013/14</th>
<th>2014/15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of students measured</td>
<td>33</td>
<td></td>
</tr>
<tr>
<td>Assessment score</td>
<td>3.43</td>
<td></td>
</tr>
<tr>
<td>% ratings with Met or Exceeded</td>
<td>87.88%</td>
<td></td>
</tr>
<tr>
<td>% ratings with Not Met</td>
<td>3.03%</td>
<td></td>
</tr>
</tbody>
</table>

Performance results for SO-a: Met in the 2013-14 academic year.

Embedded assessment in CS 254 – Computer Organization and Assembly Language
Assessment Instrument: The students’ understanding of basic computer architecture is assessed with a comprehensive final that includes questions and problems that cover this area and others. The test typically consists of 40 multiple-choice questions and 5 free response questions. Each semester’s test is largely based on that of the previous semester, although the specific objective questions and programming questions vary. Versions of the test have been given to students nearly every semester for the past 12 years. The questions that test different aspects computer architecture are used to compute a class score for each topic of computer architecture.

Measurable Performance Criteria: Assessment score will be determined using the student grade from the corresponding section of the test as follows:

- 90-100: Outcome Exceeded
- 80-89: Outcome Met
- 70-79: Outcome Minimally Attained
- below 70: Outcome Not Attained
Student outcome A is: [Students will gain] an ability to apply knowledge of computing and mathematics appropriate to the discipline. This is assessed with a comprehensive final that includes questions and problems that cover basic computer architecture, programming language implementation, and basics of signed binary and floating point. The test includes multiple-choice questions, problems to be worked out, and programming.

Values on the test are mapped to the 4-point scale by assigning each student a performance measure depending on their percentage score. Results per year are summarized below:
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Number of Students</strong></td>
<td>57</td>
<td>27</td>
<td>33</td>
<td>29</td>
<td>32</td>
</tr>
<tr>
<td><strong>Exceeded</strong></td>
<td>10 (18%)</td>
<td>8 (30%)</td>
<td>7 (21%)</td>
<td>13 (45%)</td>
<td>6 (19%)</td>
</tr>
<tr>
<td><strong>Met</strong></td>
<td>14 (25%)</td>
<td>9 (33%)</td>
<td>7 (21%)</td>
<td>6 (20%)</td>
<td>12 (37%)</td>
</tr>
<tr>
<td><strong>Minimally Attained</strong></td>
<td>18 (31%)</td>
<td>7 (26%)</td>
<td>12 (37%)</td>
<td>3 (10%)</td>
<td>7 (22%)</td>
</tr>
<tr>
<td><strong>Not met</strong></td>
<td>15 (26%)</td>
<td>3 (11%)</td>
<td>7 (21%)</td>
<td>7 (25%)</td>
<td>7 (22%)</td>
</tr>
<tr>
<td><strong>Average Performance Measure</strong></td>
<td>2.33</td>
<td>2.81</td>
<td>2.42</td>
<td>2.86</td>
<td>2.53</td>
</tr>
<tr>
<td><strong>% ratings Met or Exceeded</strong></td>
<td>43%</td>
<td>63%</td>
<td>42%</td>
<td>65%</td>
<td>56%</td>
</tr>
</tbody>
</table>

Students are mostly achieving student outcome A although there is considerable room for improvement. In 2013/14, 26% fell below the minimum. Part of the problem seems to be very weak math knowledge in a fair number of students. Although CS 151 is a prerequisite for this course, and MATH 152 (Calculus) is a corequisite for that course, a good number of students in this course fail to learn how to do arithmetic with binary representation of numbers. I have no good explanation of this. Perhaps they have assimilated the common attitude that computer science is programming and that programming is an isolated skill.
Student Outcome (b): An ability to analyze a problem, and identify the computing requirements appropriate to its solution.

Student Outcome (b) is assessed in CS 151 and CS 153.

Embedded assessment in the programming component of CS 151 – Computer Science I
Assessment instrument: Students are required to write Java code on all three tests (test 1, test 2, and the final exam). On test 1, they write a complete program intended to assess (a) their understanding of basic program design, and (b) ability to choose and apply an appropriate control structure for the problem at hand. On test 2, students are required to write a class based on a provided specification, and use this class in an application program. On the final exam, students are required to write a complete Java programs intended to assess (a) their ability to analyze a problem and identify an appropriate design choice to solve it, (b) their object-oriented programming skills, and (c) their ability to create and process arrays of primitive data types and arrays of objects.

Measurable performance criteria: Instructor evaluations use an assessment rubric that includes assessment criteria for four levels of performance: Not Met (1.00 – 1.99), Minimally Attained (2.0 – 2.79), Met (2.8 – 3.59), and Exceeded (3.6 – 4.0). The assessment score is calculated as follows:

<table>
<thead>
<tr>
<th>Total points on tests</th>
<th>Assessment score</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>3.6</td>
</tr>
<tr>
<td>7</td>
<td>2.8</td>
</tr>
<tr>
<td>5</td>
<td>2.0</td>
</tr>
<tr>
<td>3</td>
<td>1.5</td>
</tr>
<tr>
<td>1</td>
<td>1.0</td>
</tr>
</tbody>
</table>

Actual results are summarized below:

<table>
<thead>
<tr>
<th>Academic year</th>
<th>2013/14 (based only on final exam)</th>
<th>2014/15</th>
<th>2015/16</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of students measured</td>
<td>115</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Assessment score</td>
<td>3.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>% rating with Met or Exceed</td>
<td>68%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>% rating with Not met</td>
<td>10%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Assessment results suggest that overall CLOs are met and students are prepared to take the second course in the sequence, CS 152, as well as other courses listing CS 151 as a prerequisite. It should be noted that some CS 151 topics are revisited in CS 152 (classes, interfaces) as part of CS 152 core curriculum (class hierarchies and advanced class design), which is expected to further strengthen student understanding.

Embedded assessment in the programming component of CS 153 – Computer Science III
Assessment instrument: C programming is assessed with a C Proficiency Test, which is given every semester. The test used for this course is locally produced. The test is based in part on questions from the text book and questions from past AP Computer Science tests and books of sample Java certification examinations (modified from C++ or from Java into ANSI C). The test is intended to be similar to the C tests several local employers administer to prospective employees. The test consists of about 40 multiple-choice questions and 5 free response (programming) questions. Each semester’s test is largely based on
that of the previous semester, although the specific objective questions and programming questions vary. Versions of the test have been given to students nearly every semester for the past 10 years.

**Measurable performance criteria:** Assessment score will be determined using the student grade from the corresponding section of the test as follows:

<table>
<thead>
<tr>
<th>Score Range</th>
<th>Outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>90-100</td>
<td>Outcome Exceeded</td>
</tr>
<tr>
<td>75-89</td>
<td>Outcome Met</td>
</tr>
<tr>
<td>65-74</td>
<td>Outcome Minimally Attained</td>
</tr>
<tr>
<td>below 65</td>
<td>Outcome Not Attained</td>
</tr>
</tbody>
</table>

Values on the C proficiency test are mapped into the four categories by assigning each student a performance measure depending on their percentage score. Results per year are summarized below:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Students</td>
<td>39</td>
<td>38</td>
<td>25</td>
<td>21</td>
</tr>
<tr>
<td>Exceeded</td>
<td>13 (33%)</td>
<td>8 (21%)</td>
<td>2 (8%)</td>
<td>3 (14%)</td>
</tr>
<tr>
<td>Met</td>
<td>17 (44%)</td>
<td>20 (52%)</td>
<td>10 (40%)</td>
<td>7 (33%)</td>
</tr>
<tr>
<td>Minimally Attained</td>
<td>7 (18%)</td>
<td>8 (21%)</td>
<td>4 (16%)</td>
<td>6 (28%)</td>
</tr>
<tr>
<td>Not met</td>
<td>2 (5%)</td>
<td>2 (5%)</td>
<td>9 (36%)</td>
<td>5 (24%)</td>
</tr>
<tr>
<td>Average Performance Measure</td>
<td>3.05</td>
<td>3.00</td>
<td>2.2</td>
<td>2.38</td>
</tr>
<tr>
<td>% ratings Met or Exceeded</td>
<td>77%</td>
<td>73%</td>
<td>38%</td>
<td>47%</td>
</tr>
</tbody>
</table>

Students are mostly achieving student outcome B. Only 5% in 2013/14 fell below the minimum. However, there is plenty of room for improvement. At least part of the problem seems to be that a fair number of students seem poorly prepared for this class. Early assignments reveal poor understanding of programming logic, which is only slowly remediated as the course progresses.
Student Outcome (c): An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs

Student Outcome (c) is assessed using the following instruments:

**Assessment rubric for student programming projects in CS 253 – Data Structures**
Assessment instrument: Students are required to design and implement four programming projects according to provided sets of requirements, and subsequently evaluate their behavior to verify that those requirements are met. Projects 1 and 2 evaluate and compare a broad group of sorting algorithms in terms of their run-time and memory utilization characteristics. Project 3 implements a parsing algorithm, and project 4 requires students to build a small database utilizing a general tree to add, delete, and search for specific data. All projects require a written report to describe and evaluate the design choices made by the student and obtained results. All projects are graded on a 3-points scale for a total of 12 points.

Measurable performance criteria: Instructor evaluations uses an assessment rubric that includes assessment criteria for four levels of performance: Not Met (1.00 – 1.99), Minimally Attained (2.0 – 2.79), Met (2.8 – 3.59), and Exceeded (3.6 – 4.0). The assessment score is calculated as follows:

<table>
<thead>
<tr>
<th>Total points</th>
<th>Assessment score</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>4</td>
</tr>
<tr>
<td>10</td>
<td>3.6</td>
</tr>
<tr>
<td>8</td>
<td>2.8</td>
</tr>
<tr>
<td>6</td>
<td>2.0</td>
</tr>
<tr>
<td>4</td>
<td>1.5</td>
</tr>
<tr>
<td>2</td>
<td>1.0</td>
</tr>
</tbody>
</table>

Assessment results are summarized below:

<table>
<thead>
<tr>
<th>Academic year</th>
<th>2013/14</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of students measured</td>
<td>61</td>
</tr>
<tr>
<td>Assessment score</td>
<td>3.1</td>
</tr>
<tr>
<td>% ratings with Met or Exceed</td>
<td>73%</td>
</tr>
<tr>
<td>% ratings with Not Met</td>
<td>18%</td>
</tr>
</tbody>
</table>

**Embedded assessment in the student project component in CS 354 – Digital Systems Design**
Assessment Instrument: The students taking the Digital Design course (CS354) have to complete three projects requiring the design and implementation of digital systems by given specifications. The first project requires designing, simulating and testing a number of digital circuits by using a digital simulator and Hardware Description Language (Verilog HDL). The second project is about designing a 4-bit Arithmetic Logic Unit (ALU). The final project requires designing and testing a complete 4-bit Central Processing Unit (CPU) by using the ALU and other basic combinational circuits designed in the previous two projects. All three projects require analysis of existing solutions and making design choices to meet desired needs. Students also have to simulate and test their solutions in order to demonstrate that they meet the specifications.

Measurable Performance Criteria: Each project is graded on a 100% scale and the weighted average of the student score is calculated as Project1*0.17 + Project2*0.33 + Project3*0.5. This score is then converted into a 4-point system and used to assess the Student Outcome c (SO c) as follows: Not Met (1.00-1.99), Minimally Attained (2.00-2.79), Met (2.80-3.59), Exceeded (3.60-4.00).

Course assessment results using the new CQI Process will be available beginning in the Fall 2014 semester.
Student Outcome (d): An ability to function effectively on teams to accomplish a common goal.

Student Outcome (d) is assessed using the following instruments:

Course project assessment rubrics in CS 410 – Software Engineering
Assessment Instrument: Students in CS 410 are required to complete a substantial course project. While working on the project, students use the knowledge and skills obtained in this course covering many if not all of the course topics. Working in teams of 3-5 persons, students follow an iterative software process to design, implement and document a software system. The teamwork aspect of the course project is graded using two rubrics. The instructor completes one rubric, while the students complete the other rubric to assess their teammates. Each rubric includes such performance criteria as workload, oral and written communication skills, willingness to take initiative, creativity and ideas, cooperation, reliability, and leadership.

Measurable Performance Criteria: Each criterion on the teamwork assessment rubrics is graded on the 4-point scale where Poor = 0, Fair = 1, Average = 2, Good = 3, Excellent = 4. The score on the scale of 1 to 4 for each rubric is calculated by averaging the grades on each criterion on the respective rubric.
Assessment score is calculated using the as follows:

Assessment score (1 to 4) = (Instructor score) * 0.7 + (Student score) * 0.3

Course assessment results using the new CQI Process will be available beginning in the Fall 2014 semester.

Course project assessment rubrics in CS 498 – Senior Project
Assessment Instrument: Students in CS 498 are required to complete a substantial course project serving as a capstone in this undergraduate program. While working on a large-scale project, students use the knowledge and skills obtained during their undergraduate studies in this Computer Science program. Working in teams of 3-5 persons, students follow an iterative software process to design, implement and document a software system working in direct contact with an external client. The teamwork aspect of the course project is graded using two rubrics. The instructor completes one rubric, while the students complete the other rubric to assess their teammates. Each rubric includes such performance criteria as workload, oral and written communication skills, willingness to take initiative, creativity and ideas, cooperation, reliability, and leadership.

Measurable Performance Criteria: Each criterion on the teamwork assessment rubrics is graded on the 4-point scale where Poor = 0, Fair = 1, Average = 2, Good = 3, Excellent = 4. The score on the scale of 1 to 4 for each rubric is calculated by averaging the grades on each criterion on the respective rubric.
Assessment score is calculated using the as follows:

Assessment score (1 to 4) = (Instructor score) * 0.7 + (Student score) * 0.3

Course assessment results using the new CQI Process will be available beginning in the Spring 2015 semester.
Student Outcome (e): An understanding of professional, ethical, and social responsibilities.

Student Outcome (e) is assessed using the following instruments:
**Embedded assessment in tests in CS 410 – Software Engineering**
Assessment Instrument: Students are required to take three tests in CS 410: two midterms and a final. Five problems distributed among these tests cover student understanding of professional, ethical, and social responsibilities of a computing professional working in the software industry. Most of these questions are structured as case studies that present a real world situation where students are asked to suggest possible solutions involving the analysis of one or more possible scenarios and/or outcomes. 
**Measurable Performance Criteria:** Student grades on the five problems are averaged and converted to the 1 to 4 point scale. That score is used to assess the Student Outcome (e) as follows: Not Met (1.00-1.99), Minimally Attained (2.00-2.79), Met (2.80-3.59), Exceeded (3.60-4.00).

Course assessment results using the new CQI Process will be available beginning in the Fall 2014 semester.

**Student research scoring rubric in CS 498 – Senior Project**
Assessment Instrument: As a part of the course requirements of CS 498, students are required to write a research paper reflecting on one or more topics related to the professional, ethical, and social responsibilities of a software engineering professional. Readings required for this project include Software Engineering Code of Ethics and Professional Practice as recommended by the ACM/IEEE-CS Joint Task Force on Software Engineering Ethics and Professional Practices.
**Measurable Performance Criteria:** Student work on the research paper is cored using a specially developed rubric resulting in a single score on the 1 to 4 point scale. That score is used to assess the Student Outcome (e) as follows: Not Met (1.00-1.99), Minimally Attained (2.00-2.79), Met (2.80-3.59), Exceeded (3.60-4.00).

Course assessment results using the new CQI Process will be available beginning in the Spring 2015 semester.
Student Outcome (f): An ability to communicate effectively.

Student Outcome (f) is assessed using the following instruments:

Assessment rubric for the student research project in CS 253 – Data Structures
Assessment instrument: Students are required to research in-depth a graph algorithm of their choice to address a specific application problem. The objectives of this project are: (a) to study at least two design choices suitable for the selected application, evaluate and compare them to choose the better one, (b) implement the algorithm of their choice in the selected application context, (c) write a paper describing the results of their research, and (d) make a 10 – 12 minutes presentation in class to demonstrate and discuss their work. The project is assessed using a specially developed rubric, which assigns scores to the paper, implementation, and class presentation for a total of 20 points.

Measurable performance criteria: Instructor evaluations use an assessment rubric that includes assessment criteria for four levels of performance: Not Met (1.00 – 1.99), Minimally Attained (2.0 – 2.79), Met (2.8 – 3.59), and Exceeded (3.6 – 4.0). The assessment score is calculated as follows:

<table>
<thead>
<tr>
<th>Total points</th>
<th>Assessment score</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>4</td>
</tr>
<tr>
<td>18</td>
<td>3.6</td>
</tr>
<tr>
<td>15</td>
<td>2.8</td>
</tr>
<tr>
<td>12</td>
<td>2.0</td>
</tr>
<tr>
<td>10</td>
<td>1.5</td>
</tr>
<tr>
<td>1</td>
<td>1.0</td>
</tr>
</tbody>
</table>

Assessment results are summarized below:

<table>
<thead>
<tr>
<th>Academic year</th>
<th>2013/14</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of students measured</td>
<td>61</td>
</tr>
<tr>
<td>Assessment score</td>
<td>3.5</td>
</tr>
<tr>
<td>% ratings with Met or Exceeded</td>
<td>80%</td>
</tr>
<tr>
<td>% ratings with Not Met</td>
<td>15%</td>
</tr>
</tbody>
</table>

Assessment rubric for student projects in CS 385 – Computer Architecture
Assessment Instrument: The students taking the Computer Architecture course (CS385) have to complete a semester project to design and implement a MIPS processor from given specifications. The machine must be implemented in Hardware Description Language (Icarus Verilog HDL), tested with a simple MIPS program and properly documented. The semester project must be a teamwork of two or three students and requires writing three progress reports, a final report and making a presentation. While working of the project the students in the team need to communicate effectively in order to set up their individual goals coordinate their work and combine successfully the results in the final design of the MIPS machine. They also need to put together detailed reports clearly explaining the design, implementation, and testing of their product. Finally each student from the team prepares a set of slides and makes a presentation of his/her contribution to the final result before the class at the end of the course.

Measurable Performance Criteria: Each of the three progress reports, the final report and the student presentation (both slides and oral presentation) is graded on a 100% scale and the student score is calculated as an average of the five grades. This score is then converted into a 4-point system and used to assess the Student Outcome f (SO f) as follows: Not Met (1.00 – 1.99), Minimally Attained (2.0 – 2.79), Met (2.8 – 3.59), Exceeded (3.6 – 4.0).

Performance results for SO (f): Extent attained: Exceeded in the 2013/14 academic year.

<table>
<thead>
<tr>
<th>Academic year</th>
<th>2013/14</th>
<th>2014/15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of students measured</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td>Assessment score</td>
<td>3.7</td>
<td></td>
</tr>
<tr>
<td>% ratings with Met or Exceeded</td>
<td>85%</td>
<td></td>
</tr>
<tr>
<td>% ratings with Not Met</td>
<td>0%</td>
<td></td>
</tr>
</tbody>
</table>
**Student Outcome (g):** An ability to analyze the impact of computing on individuals, organizations, and society, including ethical, legal, security, and global policy issues

Student Outcome (g) is assessed using the following instruments:

**Embedded assessment in tests in CS 355 – Systems Programming**

Assessment Instrument: CS 355 includes the coverage of Linux operating systems, which forms the basis for many applications of computing ranging from powering the backends of most websites to running in the core of Android mobile devices. Three problems are distributed among the three tests in CS 355 cover student understanding of the rapidly evolving role of Linux OS, its many applications, and their impact on individuals, organizations, and society.

Measurable Performance Criteria: Student grades on the three embedded problems are averaged and converted to the 1 to 4 point scale. That score is used to assess the Student Outcome (e) as follows: Not Met (1.00-1.99), Minimally Attained (2.00-2.79), Met (2.80-3.59), Exceeded (3.60-4.00).

<table>
<thead>
<tr>
<th>Academic year</th>
<th>2013/14</th>
<th>2014/15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of students measured</td>
<td>22</td>
<td></td>
</tr>
<tr>
<td>Assessment score</td>
<td>3.39</td>
<td></td>
</tr>
<tr>
<td>% ratings with Met or Exceeded</td>
<td>73%</td>
<td></td>
</tr>
<tr>
<td>% ratings with Not Met</td>
<td>0%</td>
<td></td>
</tr>
</tbody>
</table>

Performance results for SO-g: Extent attained: **Met** in the 2013-14 academic year.

**Student research scoring rubric in CS 498 – Senior Project**

Assessment Instrument: As a part of the course requirements of CS 498, students are required to write a research paper, in which they reflect on the topics related to the impact of computing on individuals, organizations, and society.

Measurable Performance Criteria: Student work on the research paper is cored using a specially developed rubric resulting in a single score on the 1 to 4 point scale. That score is used to assess the Student Outcome (e) as follows: Not Met (1.00-1.99), Minimally Attained (2.00-2.79), Met (2.80-3.59), Exceeded (3.60-4.00).

Course assessment results using the new CQI Process will be available beginning in the Spring 2015 semester.
Student Outcome (h): Recognition of the need for and an ability to engage in continuing professional development.

Student Outcome (h) is assessed using the following instruments:

**Embedded assessment in tests in CS 410 – Software Engineering**

Assessment Instrument: Students are required to take three tests in CS 410: two midterms and a final. Three problems distributed among these tests cover student understanding of the need to engage in continuing professional development for any professional working in the field of computing. These questions are structured as case studies that present a real world situation where students are asked to suggest possible solutions involving the analysis of one or more possible scenarios and/or outcomes.

**Measurable Performance Criteria**: Student grades on the three problems are averaged and converted to the 1 to 4 point scale. That score is used to assess the Student Outcome (h) as follows: Not Met (1.00-1.99), Minimally Attained (2.00-2.79), Met (2.80-3.59), Exceeded (3.60-4.00).

Course assessment results using the new CQI Process will be available beginning in the Fall 2014 semester.

**Special scoring rubric in CS 498 – Senior Project**

Assessment Instrument: Students in CS 498 are required to complete a substantial course project serving as a capstone in this undergraduate program. As a part of the course project requirements, students are expected to identify and become proficient with one or more new programming language, development environment, modeling tool, etc., with which they were not familiar previously, and actively use it throughout the project. The professional development aspect of the course project is graded using two rubrics. The instructor completes one rubric, while the students complete the other rubric to assess themselves. Each rubric includes such criteria as recognizing the need for professional development, choice of development area, applicability, mastery, and usefulness for future projects.

**Measurable Performance Criteria**: Each criterion on the teamwork assessment rubrics is graded on the 4-point scale where Poor = 0, Fair = 1, Average = 2, Good = 3, Excellent = 4. The score on the scale of 1 to 4 for each rubric is calculated by averaging the grades on each criterion on the respective rubric.

Assessment score is calculated using the as follows:

\[
\text{Assessment score (1 to 4)} = (\text{Instructor score}) \times 0.7 + (\text{Student score}) \times 0.3
\]

Course assessment results using the new CQI Process will be available beginning in the Spring 2015 semester.
Student Outcome (i): An ability to use current techniques, skills, and tools necessary for computing practice.

Student Outcome (i) is assessed using the following instruments:

Course project assessment rubric in CS 410 – Software Engineering
Assessment Instrument: Students in CS 410 are required to complete a substantial course project. While working on the project, students need to use a broad range of current techniques, skills, and tools necessary for the successful completion of the course project. Students are expected to have the proper skills to be proficient with such tools as an integrated development environment (IDE) for a modern programming language, a UML modeling tool, and a version control system. The skills and tools aspect of the course project is graded using an assessment rubric.
Measurable Performance Criteria: Each skill and tool criterion on the assessment rubric is graded on the 4-point scale where Poor = 0, Fair = 1, Average = 2, Good = 3, Excellent = 4. The score on the scale of 1 to 4 for each rubric is calculated by averaging the grades on each criterion.

Course assessment results using the new CQI Process will be available beginning in the Fall 2014 semester.

Course project assessment rubric in CS 498 – Senior Project
Assessment Instrument: Students in CS 498 are required to complete a substantial course project. While working on the project, students need to use a broad range of current techniques, skills, and tools necessary for the successful completion of the course project. Students are expected to have the proper skills to be proficient with such tools as an integrated development environment (IDE) for a modern programming language, a UML modeling tool, and a version control system. The skills and tools aspect of the course project is graded using an assessment rubric.
Measurable Performance Criteria: Each skill and tool criterion on the assessment rubric is graded on the 4-point scale where Poor = 0, Fair = 1, Average = 2, Good = 3, Excellent = 4. The score on the scale of 1 to 4 for each rubric is calculated by averaging the grades on each criterion.

Course assessment results using the new CQI Process will be available beginning in the Spring 2015 semester.
**Student Outcome (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.

Student Outcome (j) is assessed using the following instruments:

**Embedded assessment in student exams in CS 385 – Computer Architecture**

Assessment Instrument: Students taking the Computer Architecture course learn fundamental concepts of Computer Science as instruction set architectures, computer arithmetic, memory systems and parallel processing, and the way these concepts are applied to develop efficient computing systems based on pipelining and multiprocessing. They also learn a modern industry-standard tool for designing computer systems, the Hardware Description Language (Verilog HDL). The students are required of take a midterm test and a final exam, which measure their level of mastering of these concepts, techniques and tools. The tests include comprehensive sets of questions covering the related course learning outcomes and are used to measure the corresponding student outcome.

Measurable Performance Criteria: Each of the midterm test and the final exam consists of 20 questions altogether covering the Computers Architecture concepts and techniques learnt in the course. The student answer to each question is graded on a 0.0-0.1 point scale. The sum of the scores from all 40 questions is then used to assess the Student Outcome j (SO j) as follows: Not Met (1.00-1.99), Minimally Attained (2.00-2.79), Met (2.80-3.59), Exceeded (3.60-4.00).

**Performance results for SO(j):** Extent attained: **Exceeded** in the 2013/14 academic year.

<table>
<thead>
<tr>
<th>Academic year</th>
<th>2013/14</th>
<th>2014/15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of students measured</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td>Assessment score</td>
<td>3.8</td>
<td></td>
</tr>
<tr>
<td>% ratings with Met or Exceeded</td>
<td>100%</td>
<td></td>
</tr>
<tr>
<td>% ratings with Not Met</td>
<td>0%</td>
<td></td>
</tr>
</tbody>
</table>

**Embedded assessment in student exams in CS 354 – Digital Systems Design**

Assessment Instrument: The Digital Design course includes a theoretical material on Boolean algebra and finite state machines, which are part of the fundamental Computer Science theory. The students learn specific techniques for analyzing and designing Boolean functions and finite state machines and mapping them into digital circuits as logic gates and memory elements. For this purpose various minimization techniques are used and students learn how to apply them and make design choices in order to create optimal solutions. The students are required of take three tests including the final exam, which measure their level of mastering the theories, concepts and techniques they learn in the Digital Design course. The final exam includes a comprehensive set of questions on the theory and problems on analysis and design of digital circuits and is used to measure the student outcome for these course learning outcomes.

Measurable Performance Criteria: The final exam consists of 10 questions and the student answer to each one is graded on a 100% scale. The average of the student scores from all questions is then converted into a 4-point system and used to assess the Student Outcome j (SO j) as follows: Not Met (1.00-1.99), Minimally Attained (2.00-2.79), Met (2.80-3.59), Exceeded (3.60-4.00).

Course assessment results using the new CQI Process will be available beginning in the Fall 2014 semester.

**Assessment rubric for the student course project in CS 355 – Systems Programming**

Assessment instrument: All students in the Systems Programming course (CS 355) are required to complete a team-based course project. The objective of the project is to design and develop a complex software system that meets a set of specified requirements with the features ranging from providing a
high-level user interface, meeting mid-level data storage considerations, to ensuring a correct low-level interaction with a hardware controller. Students are required to make design choices to balance the complexity of the system by distributing its features across various levels of architecture granularity and abstraction paradigms. The project is assessed using a specially developed rubric.

**Measurable performance criteria:** Instructor evaluations used an assessment rubric that included assessment criteria for four levels of performance: Not Met (1.00-1.99), Minimally Attained (2.00-2.79), Met (2.80-3.59), and Exceeded (3.60-4.00). The assessment score is calculated as the average student score.

<table>
<thead>
<tr>
<th>Academic year</th>
<th>2013/14</th>
<th>2014/15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of students measured</td>
<td>22</td>
<td></td>
</tr>
<tr>
<td>Assessment score</td>
<td>3.27</td>
<td></td>
</tr>
<tr>
<td>% ratings with Met or Exceeded</td>
<td>68%</td>
<td></td>
</tr>
<tr>
<td>% ratings with Not Met</td>
<td>0%</td>
<td></td>
</tr>
</tbody>
</table>

Performance results for SO-j: Extent attained: **Met** in the 2013-14 academic year.
Student Outcome (k): An ability to apply design and development principles in the construction of software systems of varying complexity

Student Outcome (k) is assessed using the following instruments:

Course project assessment rubric in CS 410 – Software Engineering
Assessment Instrument: Students in CS 410 are required to complete a substantial course project. In this project, students need to apply design and development principles in the construction of two-tiered web-based software system. The design and development aspect of the course project is graded using an assessment rubric, which includes criteria covering the categories related to system requirements, system architecture design, project planning, as well as unit, functional, integration, and system testing.
Measurable Performance Criteria: Each design and development criterion on the assessment rubric is graded on the 4-point scale where Poor = 0, Fair = 1, Average = 2, Good = 3, Excellent = 4. The score on the scale of 1 to 4 for each rubric is calculated by averaging the grades on each criterion.

Course assessment results using the new CQI Process will be available beginning in the Fall 2014 semester.

Course project assessment rubric in CS 498 – Senior Project
Assessment Instrument: Students in CS 498 are required to complete a substantial course project. In this project, students need to apply design and development principles in the construction of two-tiered web-based software system. The design and development aspect of the course project is graded using an assessment rubric, which includes criteria covering the categories related to system requirements, system architecture design, project planning, as well as unit, functional, integration, and system testing.
Measurable Performance Criteria: Each design and development criterion on the assessment rubric is graded on the 4-point scale where Poor = 0, Fair = 1, Average = 2, Good = 3, Excellent = 4. The score on the scale of 1 to 4 for each rubric is calculated by averaging the grades on each criterion.

Course assessment results using the new CQI Process will be available beginning in the Spring 2015 semester.
Section 3. Analysis.

SO (a): An ability to apply knowledge of computing and mathematics appropriate to the discipline
Current assessment results indicate that this Student Outcome is being met. There is not enough historical data to observe any student performance trends.

SO (b): An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution
Current assessment results indicate that this Student Outcome is being met. There is not enough historical data to observe any student performance trends.

SO (c): An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs
Current assessment results indicate that this Student Outcome is being met. There is not enough historical data to observe any student performance trends.

SO (d): An ability to function effectively on teams to accomplish a common goal
Course assessment results using the new CQI Process will be available beginning in the 2014/15 academic year.
There is not enough historical data to observe any student performance trends.

SO (e): An understanding of professional, ethical, and social responsibilities
Course assessment results using the new CQI Process will be available beginning in the 2014/15 academic year.
There is not enough historical data to observe any student performance trends.

SO (f): An ability to communicate effectively
Current assessment results indicate that this Student Outcome is being met. There is not enough historical data to observe any student performance trends.

SO (g): An ability to analyze the impact of computing on individuals, organizations, and society, including ethical, legal, security, and global policy issues
Current assessment results indicate that this Student Outcome is being met. There is not enough historical data to observe any student performance trends.

SO (h): Recognition of the need for and an ability to engage in continuing professional development
Course assessment results using the new CQI Process will be available beginning in the 2014/15 academic year.
There is not enough historical data to observe any student performance trends.

SO (i): An ability to use current techniques, skills, and tools necessary for computing practice
Course assessment results using the new CQI Process will be available beginning in the 2014/15 academic year.
There is not enough historical data to observe any student performance trends.

SO (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
Current assessment results indicate that this Student Outcome is being met. There is not enough historical data to observe any student performance trends.
SO (k): An ability to apply design and development principles in the construction of software systems of varying complexity
Current assessment results indicate that this Student Outcome is being met. There is not enough historical data to observe any student performance trends.
Section 4. Use of Results

SO (a): An ability to apply knowledge of computing and mathematics appropriate to the discipline
The new CQI Process has been adopted effective Spring 2014 semester and will not be completely rolled out until the end of the 2014/15 Academic Year. Current results are partial and do not indicate any need for change.

SO (b): An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution
The new CQI Process has been adopted effective Spring 2014 semester and will not be completely rolled out until the end of the 2014/15 Academic Year. Current results are partial and do not indicate any need for change.

SO (c): An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs
The new CQI Process has been adopted effective Spring 2014 semester and will not be completely rolled out until the end of the 2014/15 Academic Year. Current results are partial and do not indicate any need for change.

SO (d): An ability to function effectively on teams to accomplish a common goal
The new CQI Process has been adopted effective Spring 2014 semester and will not be completely rolled out until the end of the 2014/15 Academic Year. Current results are partial and do not indicate any need for change.

SO (e): An understanding of professional, ethical, and social responsibilities
The new CQI Process has been adopted effective Spring 2014 semester and will not be completely rolled out until the end of the 2014/15 Academic Year. Current results are partial and do not indicate any need for change.

SO (f): An ability to communicate effectively
The new CQI Process has been adopted effective Spring 2014 semester and will not be completely rolled out until the end of the 2014/15 Academic Year. Current results are partial and do not indicate any need for change.

SO (g): An ability to analyze the impact of computing on individuals, organizations, and society, including ethical, legal, security, and global policy issues
The new CQI Process has been adopted effective Spring 2014 semester and will not be completely rolled out until the end of the 2014/15 Academic Year. Current results are partial and do not indicate any need for change.

SO (h): Recognition of the need for and an ability to engage in continuing professional development
The new CQI Process has been adopted effective Spring 2014 semester and will not be completely rolled out until the end of the 2014/15 Academic Year. Current results are partial and do not indicate any need for change.

SO (i): An ability to use current techniques, skills, and tools necessary for computing practice
The new CQI Process has been adopted effective Spring 2014 semester and will not be completely rolled out until the end of the 2014/15 Academic Year. Current results are partial and do not indicate any need for change.
SO (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

The new CQI Process has been adopted effective Spring 2014 semester and will not be completely rolled out until the end of the 2014/15 Academic Year. Current results are partial and do not indicate any need for change.

SO (k): An ability to apply design and development principles in the construction of software systems of varying complexity

The new CQI Process has been adopted effective Spring 2014 semester and will not be completely rolled out until the end of the 2014/15 Academic Year. Current results are partial and do not indicate any need for change.
Section 5. General Education.

General Education Summary

CS 110 and CS 113 are offered by the department and satisfy the Skill Area II of General Education requirements.

The relevant Learning Outcomes of General Education that apply to Skill Area II and to CS 110 and CS 113 are:

QS: To strengthen quantitative skills

QS1. Apply mathematical and statistical techniques as a means of analysis within a variety of disciplines
QS2. Assess the strengths and weaknesses of these techniques of analysis.

IFC. To develop information fluency and computer literacy. Relevant outcomes include the ability to:

IFC1. Locate, evaluate, and effectively use information from a variety of sources;
IFC2. Use computers for research, analysis, and expression;
IFC3. Analyze the effects of information technology on society.
General Education Findings

Both CS 110 and CS 113 supported the following general education learning outcomes: QS 1, IFC 1 and IFC 3. Student attainment of these learning outcomes was assessed using the following instruments:

Questions embedded in midterm and final exam in CS 110:
1. Briefly explain the role of the Internet as the primary modern technology for online communication (IFC1, IFC2).
2. Name three differences and three similarities between Internet Explorer and Firefox (IFC1).
3. Briefly explain what is “email etiquette” and provide specific examples to illustrate your answer (IFC3).
4. Explain the following statement: “you have no expectation of privacy on the Internet”. Give two specific examples supporting our answer (IFC3).
5. Give an example of a reasonably complex web address (link) and explain the meaning/purpose of all of its components (IFC3).
6. Explain the rationale behind the principle “code a little, test a little” for developing web pages with HTML and programming with JavaScript (QS1).
7. Using JavaScript, declare a variable myName and initialize it with your name (QS1).
8. Write a complete code for an HTML page that displays an ordered list with the movie titles you have seen recently (QS1).
9. Write a complete code for an HTML page containing a JavaScript program that displays an alert dialog box with the words Hello, world! (QS1)

Questions embedded in midterm and final exam in CS 113:
1. Determine the output displayed in the listbox when the command button is clicked (QS1, IFC1).
   Private Sub BtnDisplay_Click(...) Handles BtnDisplay.Click
   Dim nom As String
   nom = "David"
   ListBox1.Items.Add("Hello " & nom)
   Hello(nom)
   ListBox1.Items.Add("Hello " & nom)
   End Sub

   Sub Hello(byval x As String)
   x = "Bob"
   ListBox1.Items.Add("Hello " & x)
   End Sub

2. Identify any errors (QS1).
   Private Sub BtnDisplay_Click(...) Handles BtnDisplay.Click
   Dim n As Double
   n = 5
   Alphabet()
   End Sub

   Sub Alphabet(n As Integer)
   ListBox1.Items.Add("abcdefghijklmnopqrstuvwxyz".Substring(2,n))
   End Sub

3. Determine whether the condition is true or false. Assume a = 2 and b = 3 (QS1, IFC1).
   NOT((a < b) And (a < (b + a)))
   (a = b) And (a * a < b * b)) OR ((b < a) And (2 * a < b))
   ("Duck" < "pig") And ("pig" < "big")
4. In the following question, rewrite the program using a loop (either a "Do While … Loop" or a "For…Next loop") (QS1).

Private Sub BtnDisplay_Click(…) Handles BtnDisplay.Click
    Listbox.items.add( "Hello"
    Listbox.items.add( "Hello"
    Listbox.items.add( "Hello"
    Listbox.items.add( "Hello"
End Sub

5. In the following question, what does this program do? (QS1)

Private Sub btnDisplay_Click(…) Handles BtnDisplay.Click
    Dim flag As Integer, num As Double
    flag = 0
    Do While flag <> 1
        num = Cdbl(InputBox("Enter a number"))
        If num*num<0 Then
            flag = 5
        End If
    Loop
End Sub

6. Design a form with three text boxes to get three numbers, and print the average into a list box. The result should be displayed with complete sentence by reiterating the numbers entered by user. For example, if user entered numbers 3, 4, and 8, the list box should display "The average of 3, 4, and 8 is 5." (QS1, IFC3)

7. Complete the following program to display the sum of the odd numbers from 1 to 100. You Must write a loop (either a "Do While … Loop" or a "For…Next loop") for this program. (QS1)

Private Sub BtnDisplay_Click(…) Handles BtnDisplay.Click
    Dim Sum As Integer, i As Integer
    LstOutput.items.Clear()
End Sub

8. A computer store sells diskettes at 25 cents each for small orders or at 20 cents each for orders of 100 diskettes or more. Write a program that requests the number of diskettes ordered and displays the total cost. (QS1, IFC3)

9. Write a subroutine procedure that accepts three numbers and displays the largest of them in the list box. (QS1)

Sub Max3(byVal n1 as Double, byVal n2 as Double, byVal n3 as Double)
End Sub

10. Change the Do-While loop with a For-Next loop in the program below. The new program should work and print exactly the same as the original one. (QS1)

Private Sub BtnDisplay_Click(…) Handles BtnDisplay.Click
    Dim N ,S as double
    S=0
    N=1
    Do while N<11
        S= S+N
        N = N + 1
    End Do
End Sub
Instructors of each section must embed these questions in their tests and report the tabulated data to the course coordinators.

General education learning outcomes are assessed using the corresponding embedded questions listed below labeled QS1, IFC1, and IFC3.

**General Education Analysis:**

The general education courses seem to be mostly working, but a fair number of students fall below the acceptable level. Although assessment does not show this directly, the impression of instructors is that students are poorly motivated. Students who take general education computer science often “hate math”, which means they hate goals QS 1 and QS2.

**General Education Use of Results:**

The department has addressed the problem of poor student interest and motivation by adding *LAN School* to some classrooms, by providing tutors specific to our courses, and experimenting with *Alice*.

*LAN School*. To increase interaction and (hopefully) student interest, the Computer Science Dept. bought and installed *LAN School*, a software product which supports interactive quizzes (software “clickers”). This product is now provided by the university in selected classrooms. It also enables instructors to look at each student’s computer screen from the instructor’s workstation. This makes answering student questions easier, and cuts down on distractions (students not using the computer for classwork.)

*CS Tutors*. The CS department has worked to keep open a computer lab (MS 313) during the day so that they can do work and get help that is specific to their courses. There are three CS undergraduate students working as tutors specifically for these classes. In some years the department has a graduate assistant who also works on this. This policy was started in 2010. (We occasionally had tutors in previous years, but the program was considerably expanded in that year.)

*Alice*. The software environment *Alice* is purpose built to provide an interesting interactive programming environment for beginning students. It is commonly used in grade schools and high schools. One of our full-time instructors went to a workshop on how to teach with *Alice* and used it in four sections of CS 113. Unfortunately, we don’t have any direct measure that the experiment was successful, but anecdotal evidence is that it worked. We may do more with this.
Section 6. Assessment Plan

The assessment process in the Computer Science department is driven by ABET/CAC requirements and is outlined in the CQI Process document. It fits with the 6-year ABET assessment review cycle.

1. **Where improvements are needed.**
   a. It appears that generally, the program overall, as well as individual courses, do achieve their learning objectives. However, since the CQI Process has been adopted very recently and will not take full effect until the end of the 2014/15 year, assessment results for the entire program are not yet available.

2. **Strategy.**
   a. As mentioned above, CQI Process needs to be rolled out throughout the entire program, which will be achieved by the end of 2014/15 AY.
   b. An exit interview has been added as a program requirement for all graduating students in the spring semester. It is required for students in both Honors and Alternative programs.
   c. CQI Process will be expanded to include tracking of graduates.

3. **Goal.**
   a. Based on the current assessment data, no individual Student Outcome needs improvement.
   b. Tracking of graduates, creating a number of portfolios for past graduates showcasing their success stories, improving assessment of program outcomes no covered by the major field test.

4. **Schedule.**
   a. Ongoing. As described above.

5. **Target group.**
   a. n/a

6. **Data to be collected.**
   a. Varies by each student outcome as described in the CQI Process.

7. **When data will be analyzed.**
   a. January for the previous Fall semester.
   b. September for the previous Spring semester.

8. **Approvals.**
   a. n/a

9. **Implementation of improvements.**
   a. Ongoing.
1. Industry Advisory Board

Industry Advisory Board (IAB) is comprised of the representatives of various constituencies of the program. The objective of the IAB is to help the department create and maintain the curriculum that continues to meet the changing needs of industry, and to provide feedback on the preparedness of its graduates for fulfilling current business needs.

Current membership of the IAB is as follows:

- Nasreen Ali (Priceline.com)
- Steve Bazinet (UConn Health Center)
- Tim Davoll (Computer Sciences Corporation)
- Earl DuBack (General Dynamics, retired)
- Martha Leonard (The Hartford, IT Leadership Development Program)
- Steve Mead (Tunxis Community College)
- Norma Ortega (Travelers, Talent Acquisition University Relations)
- Cliff Pelletier (CCSU CS faculty emeritus)

IAB meets once a year, typically during the Spring semester. Minutes of the IAB meetings are kept to reflect the decisions concerning the educational objectives of the program as described in Section 3.
2. Department Meetings

Department meetings are an integral part of the department's functioning. They play a vital role in keeping the program current and stable. Typically, the department holds general meetings once a month during the academic year, although special meetings focusing on curricular or assessment issues may be held as needed. Any decisions concerning any of the program elements, assessment instruments, metrics and their criteria, evaluation procedures, or any other part of the program described in this document, must be made at a department meeting. Such decisions must be documented in the minutes of the department meetings for posterity. As necessary, corresponding changes will be made to this document to reflect these decisions.
3. Program Educational Objectives

As defined in the current ABET criteria [1],

*Program educational objectives (PEO’s) are broad statements that describe what graduates are expected to attain within a few years of graduation. Program educational objectives are based on the needs of the program’s constituencies.*

Approved by the general meeting of the IAB on April 22, 2013 and amended on November 27, 2013 current PEO’s are as follows:

- **PEO 1 – fundamentals.** Graduates will apply a broad understanding of the fundamental theories, concepts, and applications of computer science in their career.
- **PEO 2 – careers.** Graduates will be engaged in a wide range of careers in computer science and information technology.
- **PEO 3 – communication.** Graduates will communicate effectively, both orally and in writing.
- **PEO 4 – life-long learning.** Graduates will engage in life-long learning or continue with graduate study in computer science.
- **PEO 5 – ethics.** Graduates will act ethically and with social responsibility in their careers.

PEO’s are regularly reviewed by the IAB and can be modified to better address the needs of the program’s constituencies. Any changes, additions, or deletions of the PEO’s must be approved by the IAB.
4. Student Outcomes

As defined in the current ABET criteria [1],

*Student outcomes (SO’s) describe what students are expected to know and be able to do by the time of graduation. These relate to the knowledge, skills, and behaviors that students acquire as they progress through the program.*

Current SO’s are as follows:

- SO (a): An ability to apply knowledge of computing and mathematics appropriate to the discipline;
- SO (b): An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution;
- SO (c): An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs;
- SO (d): An ability to function effectively on teams to accomplish a common goal;
- SO (e): An understanding of professional, ethical, and social responsibilities;
- SO (f): An ability to communicate effectively;
- SO (g): An ability to analyze the impact of computing on individuals, organizations, and society, including ethical, legal, security, and global policy issues;
- SO (h): Recognition of the need for and an ability to engage in continuing professional development;
- SO (i): An ability to use current techniques, skills, and tools necessary for computing practice;
- SO (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;
- SO (k): An ability to apply design and development principles in the construction of software systems of varying complexity.

**Table 1. Mapping of Student Outcomes to Program Educational Objectives.**

<table>
<thead>
<tr>
<th>Program Educational Objectives</th>
<th>Student outcomes</th>
</tr>
</thead>
<tbody>
<tr>
<td>PEO 1 – fundamentals</td>
<td>a</td>
</tr>
<tr>
<td>PEO 2 – careers</td>
<td>x</td>
</tr>
<tr>
<td>PEO 3 – communication</td>
<td>x</td>
</tr>
<tr>
<td>PEO 4 – life-long learning</td>
<td>x</td>
</tr>
<tr>
<td>PEO 5 – ethics</td>
<td>x</td>
</tr>
</tbody>
</table>

SO’s and their mapping to PEO’s (Table 1) are reviewed on a regular basis and approved by the department and the IAB to ensure a continued compliance with the set of ABET criteria [1]. The program ensures that graduating students meet the current SO’s using the assessment techniques and evaluation procedures outlined in Sections 5 and 6.
5. Assessment
As defined in the current ABET criteria [1],

*Assessment is one or more processes that identify, collect, and prepare data to evaluate the attainment of student outcomes.*

5.1. Program-level Assessment
Table 2 shows the relationship between the SO’s and the curricular components (courses offered in the program and individual components of the Major Field Test):

- x – a curricular component supports the corresponding SO;
- A – a curricular component is used for assessment of the corresponding SO;
- * – a rarely offered course;
- ** – a topics course that may support a variety of SO’s, depending on the offered topic.
Table 2. Mapping of student outcomes to curricular components (including courses) and assessment instruments.

<table>
<thead>
<tr>
<th>Curricular components and faculty coordinators</th>
<th>Student outcomes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>a  b  c  d  e  f  g  h  i  j  k</td>
</tr>
<tr>
<td>Core courses and requirements</td>
<td></td>
</tr>
<tr>
<td>CS 151 Zlatareva</td>
<td>x   A   x</td>
</tr>
<tr>
<td>CS 152 Pevac</td>
<td>A   x   x</td>
</tr>
<tr>
<td>CS 153 Kjell</td>
<td>x   A   x</td>
</tr>
<tr>
<td>CS 253 Zlatareva</td>
<td>x   x   A</td>
</tr>
<tr>
<td>CS 254 Kjell</td>
<td>A   x   x</td>
</tr>
<tr>
<td>CS 354 Markov</td>
<td>x   x   A</td>
</tr>
<tr>
<td>CS 355 Kurkovsky</td>
<td>x   x   x   x</td>
</tr>
<tr>
<td>CS 385 Markov</td>
<td>x   x   x   x</td>
</tr>
<tr>
<td>CS 410 Kurkovsky</td>
<td>x   x   A   x</td>
</tr>
<tr>
<td>CS 498 Kurkovsky</td>
<td>x   x   A   x</td>
</tr>
<tr>
<td>MFT</td>
<td>A</td>
</tr>
<tr>
<td>Advanced electives</td>
<td></td>
</tr>
<tr>
<td>CS 407**</td>
<td></td>
</tr>
<tr>
<td>CS 415 Kurkovsky</td>
<td>x   x   x   x</td>
</tr>
<tr>
<td>CS 423 Kjell</td>
<td>x   x   x</td>
</tr>
<tr>
<td>CS 425*</td>
<td>x   x   x</td>
</tr>
<tr>
<td>CS 460 Pevac</td>
<td>x   x   x   x</td>
</tr>
<tr>
<td>CS 462 Zlatareva</td>
<td>x   x   x</td>
</tr>
<tr>
<td>CS 463 Pevac</td>
<td>x   x   x</td>
</tr>
<tr>
<td>CS 464 Abdollahzadeh</td>
<td>x</td>
</tr>
<tr>
<td>CS 465*</td>
<td>x   x   x</td>
</tr>
<tr>
<td>CS 473*</td>
<td>x   x   x</td>
</tr>
<tr>
<td>CS 481 Abdollahzadeh</td>
<td>x   x   x</td>
</tr>
<tr>
<td>CS 483*</td>
<td>x   x</td>
</tr>
<tr>
<td>CS 490 Williams</td>
<td>x   x   x   x</td>
</tr>
<tr>
<td>CS 491*</td>
<td>x   x</td>
</tr>
<tr>
<td>CS 492 Williams</td>
<td>x   x</td>
</tr>
<tr>
<td>CS 495</td>
<td>x   x   x   x</td>
</tr>
<tr>
<td>Auxiliary</td>
<td></td>
</tr>
<tr>
<td>CS 290**</td>
<td></td>
</tr>
<tr>
<td>CS 300*</td>
<td>x   x   x   x</td>
</tr>
<tr>
<td>CS 301*</td>
<td>x   x   x   x</td>
</tr>
<tr>
<td>CS 398*</td>
<td>x   x   x</td>
</tr>
<tr>
<td>CS 499*</td>
<td>x   x   x</td>
</tr>
</tbody>
</table>

Each SO is assessed using assessment instruments from one or more curricular components. The mapping of courses to SO’s, as well as the selection of courses used for SO assessment are regularly reviewed and approved by the department.
5.2. Assessment of Student Outcomes

This section describes individual instruments and criteria used to measure the attainment level of each SO. The instruments include course-embedded assessment and the results of the Major Field Test (MFT) that all graduating seniors take in the Spring semester of their final year. The data from each assessment instrument becomes available when the corresponding course-level assessment is completed (typically, in January for the previous Fall semester and in September for the previous Spring semester). All courses with embedded assessment are required for all students in the program. They are offered either every semester or once a year. Therefore, a complete new set of assessment data typically becomes available once a year.

The data collected as a result of the assessment process is evaluated as described in Section 6 to make decisions leading to changes in the program. For this purpose, the following qualitative metrics have been established for each outcome:

- 3.60 to 4.00: Exceeded;
- 2.80 to 3.59: Met;
- 2.00 to 2.79: Minimally Attained;
- 1.00 to 1.99: Not Met.

An assessment score on the scale of 1 to 4 is calculated for each assessment instrument described below and evaluated by placing it into one of the corresponding qualitative metrics.
5.2.1 SO a: An ability to apply knowledge of computing and mathematics appropriate to the discipline

Student Outcome (a) is assessed using the following instruments:

5.2.1.1 Embedded assessment in CS 152 – Computer Science II
Assessment Instrument: Students taking the course reinforce and extend the knowledge of lists including arrays of objects and list implemented with ArrayLists. In addition, they learn fundamental object-oriented concepts such as inheritance, polymorphism via inheritance, and polymorphism via interfaces. They also learn interfaces, abstract classes, input and output files, protected visibility modifier, the use of super reference in methods and in constructor, and exceptions. Parameter passing knowledge is extended by including formal parameters of interface type, and abstract class type and actual parameters as objects from the class that implements the interface or from descendants of such classes. Additional topics covered in the course include linear and binary searching, and selection and insertion sorting, recursion, basic GUI components and containers, and event driven programming. Starting in the Spring of 2014 linked lists will also be covered in CS152. The students are required to implement five to six projects, and to take two tests. The projects, tests questions and final exam questions are covering the related course learning outcomes and are used to measure the corresponding student outcomes. At the end of the semester students take comprehensive final exam which is the best instrument to measure their final level of mastering the concepts learned in the course.
Measurable Performance Criteria: The final exam is comprehensive and reflects what students have learned, after implementing five to six projects, and after taking two tests during the semester. The average score of the final exams is converted from 0 to 100 point scale into 1 to 4 point scale. That score is used to assess the Student Outcome (a) as follows: Not Met (1.00-1.99), Minimally Attained (2.00-2.79), Met (2.80-3.59), Exceeded (3.60-4.00).

5.2.1.2 Embedded assessment in CS 254 – Computer Organization and Assembly Language
Assessment Instrument: The students’ understanding of basic computer architecture is assessed with a comprehensive final that includes questions and problems that cover this area and others. The test typically consists of 40 multiple-choice questions and 5 free response questions. Each semester’s test is largely based on that of the previous semester, although the specific objective questions and programming questions vary. Versions of the test have been given to students nearly every semester for the past 12 years. The questions that test different aspects computer architecture are used to compute a class score for each topic of computer architecture.
Measurable Performance Criteria: Assessment score will be determined using the student grade from the corresponding section of the test as follows:

- 90-100 Outcome Exceeded
- 80-89 Outcome Met
- 70-79 Outcome Minimally Attained
- below 70 Outcome Not Attained

5.2.1.3 Major Field Test indicator “Programming and Software Engineering”
Assessment instrument: All graduating seniors are required to complete the Major Field Test (MFT) in the Spring of their final year to test their knowledge across a comprehensive set of topics covered by the entire program. The test consists of 66 questions and lasts 120 minutes. MFT results are comprised of three assessment indicators, one of which is titled “Programming and Software Engineering.”
Measurable performance criteria: The objective level of difficulty of this test varies from year to year; however, all students taking the test nationwide receive the same set of questions. Therefore, the percentile of correct answers for the student group taking the test during a given semester is used to assess student performance on this test with the following qualitative metrics: 0% to 24% = Not Met (1.00-1.99), 25% to 44% = Minimally Attained (2.00-2.79), 45% to 64% = Met (2.80-3.59), and 65% to 100% = Exceeded (3.60-4.00). The assessment score is calculated as:
\[ \text{Assessment score (1 to 4)} = \frac{\text{percentile}}{25} + 1.0 \]
5.2.2 SO b: An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution

Student Outcome (b) is assessed using the following instruments:

5.2.2.1 Embedded assessment in the programming component of CS 151 – Computer Science I

Assessment instrument: Students are required to write Java code on all three tests (test 1, test 2, and the final exam). On test 1, they write a complete program intended to assess (a) their understanding of basic program design, and (b) ability to choose and apply an appropriate control structure for the problem at hand. On test 2, students are required to write a class based on a provided specification, and use this class in an application program. On the final exam, students are required to write a complete Java programs intended to assess (a) their ability to analyze a problem and identify an appropriate design choice to solve it, (b) their object-oriented programming skills, and (c) their ability to create and process arrays of primitive data types and arrays of objects.

Measurable performance criteria: Instructor evaluations use an assessment rubric that includes assessment criteria for four levels of performance: Not Met (1.00 – 1.99), Minimally Attained (2.0 – 2.79), Met (2.8 – 3.59), and Exceeded (3.6 – 4.0). The assessment score is calculated as follows:

<table>
<thead>
<tr>
<th>Total points on tests</th>
<th>Assessment score</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>3.6</td>
</tr>
<tr>
<td>7</td>
<td>2.8</td>
</tr>
<tr>
<td>5</td>
<td>2.0</td>
</tr>
<tr>
<td>3</td>
<td>1.5</td>
</tr>
<tr>
<td>1</td>
<td>1.0</td>
</tr>
</tbody>
</table>

5.2.2.2 Embedded assessment in the programming component of CS 153 – Computer Science III

Assessment instrument: C programming is assessed with a C Proficiency Test, which is given every semester. The test used for this course is locally produced. The test is based in part on questions from the text book and questions from past AP Computer Science tests and books of sample Java certification examinations (modified from C++ or from Java into ANSI C). The test is intended to be similar to the C tests several local employers administer to prospective employees. The test consists of about 40 multiple-choice questions and 5 free response (programming) questions. Each semester's test is largely based on that of the previous semester, although the specific objective questions and programming questions vary. Versions of the test have been given to students nearly every semester for the past 10 years.

Measurable performance criteria: Assessment score will be determined using the student grade from the corresponding section of the test as follows:

<table>
<thead>
<tr>
<th>90-100</th>
<th>Outcome Exceeded</th>
</tr>
</thead>
<tbody>
<tr>
<td>75-89</td>
<td>Outcome Met</td>
</tr>
<tr>
<td>65-74</td>
<td>Outcome Minimally Attained</td>
</tr>
<tr>
<td>below 65</td>
<td>Outcome Not Attained</td>
</tr>
</tbody>
</table>
5.2.3 SO c: An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs

Student Outcome (c) is assessed using the following instruments:

5.2.3.1 Assessment rubric for student programming projects in CS 253 – Data Structures

Assessment Instrument: Students are required to design and implement four programming projects according to provided sets of requirements, and subsequently evaluate their behavior to verify that those requirements are met. Projects 1 and 2 evaluate and compare a broad group of sorting algorithms in terms of their run-time and memory utilization characteristics. Project 3 implements a parsing algorithm, and project 4 requires students to build a small database utilizing a general tree to add, delete, and search for specific data. All projects require a written report to describe and evaluate the design choices made by the student and obtained results. All projects are graded on a 3-points scale for a total of 12 points.

Measurable Performance Criteria: Instructor evaluations uses an assessment rubric that includes assessment criteria for four levels of performance: Not Met (1.00 – 1.99), Minimally Attained (2.0 – 2.79), Met (2.8 – 3.59), and Exceeded (3.6 – 4.0). The assessment score is calculated as follows:

<table>
<thead>
<tr>
<th>Total points</th>
<th>Assessment score</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>4</td>
</tr>
<tr>
<td>10</td>
<td>3.6</td>
</tr>
<tr>
<td>8</td>
<td>2.8</td>
</tr>
<tr>
<td>6</td>
<td>2.0</td>
</tr>
<tr>
<td>4</td>
<td>1.5</td>
</tr>
<tr>
<td>2</td>
<td>1.0</td>
</tr>
</tbody>
</table>

5.2.3.2 Embedded assessment in the student project component in CS 354 – Digital Systems Design

Assessment Instrument: The students taking the Digital Design course (CS354) have to complete three projects requiring the design and implementation of digital systems by given specifications. The first project requires designing, simulating and testing a number of digital circuits by using a digital simulator and Hardware Description Language (Verilog HDL). The second project is about designing a 4-bit Arithmetic Logic Unit (ALU). The final project requires designing and testing a complete 4-bit Central Processing Unit (CPU) by using the ALU and other basic combinational circuits designed in the previous two projects. All three projects require analysis of existing solutions and making design choices to meet desired needs. Students also have to simulate and test their solutions in order to demonstrate that they meet the specifications.

Measurable Performance Criteria: Each project is graded on a 100% scale and the weighted average of the student score is calculated as Project1*0.17 + Project2*0.33 + Project3*0.5. This score is then converted into a 4-point system and used to assess the Student Outcome c (SO c) as follows: Not Met (1.00-1.99), Minimally Attained (2.00-2.79), Met (2.80-3.59), Exceeded (3.60-4.00).
5.2.4  SO d: An ability to function effectively on teams to accomplish a common goal

Student Outcome (d) is assessed using the following instruments:

5.2.4.1  Course project assessment rubrics in CS 410 – Software Engineering

Assessment Instrument: Students in CS 410 are required to complete a substantial course project. While working on the project, students use the knowledge and skills obtained in this course covering many if not all of the course topics. Working in teams of 3-5 persons, students follow an iterative software process to design, implement and document a software system. The teamwork aspect of the course project is graded using two rubrics. The instructor completes one rubric, while the students complete the other rubric to assess their teammates. Each rubric includes such performance criteria as workload, oral and written communication skills, willingness to take initiative, creativity and ideas, cooperation, reliability, and leadership.

Measurable Performance Criteria: Each criterion on the teamwork assessment rubrics is graded on the 4-point scale where Poor = 0, Fair = 1, Average = 2, Good = 3, Excellent = 4. The score on the scale of 1 to 4 for each rubric is calculated by averaging the grades on each criterion on the respective rubric.

Assessment score is calculated using the as follows:

\[ \text{Assessment score (1 to 4)} = (\text{Instructor score}) \times 0.7 + (\text{Student score}) \times 0.3 \]

5.2.4.2  Course project assessment rubrics in CS 498 – Senior Project

Assessment Instrument: Students in CS 498 are required to complete a substantial course project serving as a capstone in this undergraduate program. While working on a large-scale project, students use the knowledge and skills obtained during their undergraduate studies in this Computer Science program. Working in teams of 3-5 persons, students follow an iterative software process to design, implement and document a software system working in direct contact with an external client. The teamwork aspect of the course project is graded using two rubrics. The instructor completes one rubric, while the students complete the other rubric to assess their teammates. Each rubric includes such performance criteria as workload, oral and written communication skills, willingness to take initiative, creativity and ideas, cooperation, reliability, and leadership.

Measurable Performance Criteria: Each criterion on the teamwork assessment rubrics is graded on the 4-point scale where Poor = 0, Fair = 1, Average = 2, Good = 3, Excellent = 4. The score on the scale of 1 to 4 for each rubric is calculated by averaging the grades on each criterion on the respective rubric.

Assessment score is calculated using the as follows:

\[ \text{Assessment score (1 to 4)} = (\text{Instructor score}) \times 0.7 + (\text{Student score}) \times 0.3 \]
5.2.5  **SO e: An understanding of professional, ethical, and social responsibilities**  
Student Outcome (e) is assessed using the following instruments:

**5.2.5.1 Embedded assessment in tests in CS 410 – Software Engineering**  
**Assessment Instrument:** Students are required to take three tests in CS 410: two midterms and a final. Five problems distributed among these tests cover student understanding of professional, ethical, and social responsibilities of a computing professional working in the software industry. Most of these questions are structured as case studies that present a real world situation where students are asked to suggest possible solutions involving the analysis of one or more possible scenarios and/or outcomes.  
**Measurable Performance Criteria:** Student grades on the five problems are averaged and converted to the 1 to 4 point scale. That score is used to assess the Student Outcome (e) as follows: Not Met (1.00-1.99), Minimally Attained (2.00-2.79), Met (2.80-3.59), Exceeded (3.60-4.00).

**5.2.5.2 Student research scoring rubric in CS 498 – Senior Project**  
**Assessment Instrument:** As a part of the course requirements of CS 498, students are required to write a research paper reflecting on one or more topics related to the professional, ethical, and social responsibilities of a software engineering professional. Readings required for this project include Software Engineering Code of Ethics and Professional Practice as recommended by the ACM/IEEE-CS Joint Task Force on Software Engineering Ethics and Professional Practices.  
**Measurable Performance Criteria:** Student work on the research paper is cored using a specially developed rubric resulting in a single score on the 1 to 4 point scale. That score is used to assess the Student Outcome (e) as follows: Not Met (1.00-1.99), Minimally Attained (2.00-2.79), Met (2.80-3.59), Exceeded (3.60-4.00).
5.2.6 SO f: An ability to communicate effectively
Student Outcome (f) is assessed using the following instruments:

5.2.6.1 Assessment rubric for the student research project in CS 253 – Data Structures
Assessment instrument: Students are required to research in-depth a graph algorithm of their choice to address a specific application problem. The objectives of this project are: (a) to study at least two design choices suitable for the selected application, evaluate and compare them to choose the better one, (b) implement the algorithm of their choice in the selected application context, (c) write a paper describing the results of their research, and (d) make a 10 – 12 minutes presentation in class to demonstrate and discuss their work. The project is assessed using a specially developed rubric, which assigns scores to the paper, implementation, and class presentation for a total of 20 points.
Measurable performance criteria: Instructor evaluations use an assessment rubric that includes assessment criteria for four levels of performance: Not Met (1.00 – 1.99), Minimally Attained (2.0 – 2.79), Met (2.8 – 3.59), and Exceeded (3.6 – 4.0). The assessment score is calculated as follows:

<table>
<thead>
<tr>
<th>Total points</th>
<th>Assessment score</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>4</td>
</tr>
<tr>
<td>18</td>
<td>3.6</td>
</tr>
<tr>
<td>15</td>
<td>2.8</td>
</tr>
<tr>
<td>12</td>
<td>2.0</td>
</tr>
<tr>
<td>10</td>
<td>1.5</td>
</tr>
<tr>
<td>1</td>
<td>1.0</td>
</tr>
</tbody>
</table>

5.2.6.2 Assessment rubric for student projects in CS 385 – Computer Architecture
Assessment Instrument: The students taking the Computer Architecture course (CS385) have to complete a semester project to design and implement a MIPS processor from given specifications. The machine must be implemented in Hardware Description Language (Icarus Verilog HDL), tested with a simple MIPS program and properly documented. The semester project must be a teamwork of two or three students and requires writing three progress reports, a final report and making a presentation. While working of the project the students in the team need to communicate effectively in order to set up their individual goals coordinate their work and combine successfully the results in the final design of the MIPS machine. They also need to put together detailed reports clearly explaining the design, implementation, and testing of their product. Finally each student from the team prepares a set of slides and makes a presentation of his/her contribution to the final result before the class at the end of the course.
Measurable Performance Criteria: Each of the three progress reports, the final report and the student presentation (both slides and oral presentation) is graded on a 100% scale and the student score is calculated as an average of the five grades. This score is then converted into a 4-point system and used to assess the Student Outcome f (SO f) as follows: Not Met (1.00-1.99), Minimally Attained (2.00-2.79), Met (2.80-3.59), Exceeded (3.60-4.00).
5.2.7 SO g: An ability to analyze the impact of computing on individuals, organizations, and society, including ethical, legal, security, and global policy issues

Student Outcome (g) is assessed using the following instruments:

5.2.7.1 Embedded assessment in tests in CS 355 – Systems Programming

**Assessment Instrument:** CS 355 includes the coverage of Linux operating systems, which forms the basis for many applications of computing ranging from powering the back ends of most websites to running in the core of Android mobile devices. Three problems are distributed among the three tests in CS 355 cover student understanding of the rapidly evolving role of Linux OS, its many applications, and their impact on individuals, organizations, and society.

**Measurable Performance Criteria:** Student grades on the three embedded problems are averaged and converted to the 1 to 4 point scale. That score is used to assess the Student Outcome (e) as follows: Not Met (1.00-1.99), Minimally Attained (2.00-2.79), Met (2.80-3.59), Exceeded (3.60-4.00).

5.2.7.2 Student research scoring rubric in CS 498 – Senior Project

**Assessment Instrument:** As a part of the course requirements of CS 498, students are required to write a research paper, in which they reflect on the topics related to the impact of computing on individuals, organizations, and society.

**Measurable Performance Criteria:** Student work on the research paper is cored using a specially developed rubric resulting in a single score on the 1 to 4 point scale. That score is used to assess the Student Outcome (e) as follows: Not Met (1.00-1.99), Minimally Attained (2.00-2.79), Met (2.80-3.59), Exceeded (3.60-4.00).
5.2.8 SO h: Recognition of the need for and an ability to engage in continuing professional development

Student Outcome (h) is assessed using the following instruments:

5.2.8.1 Embedded assessment in tests in CS 410 – Software Engineering

Assessment Instrument: Students are required to take three tests in CS 410: two midterms and a final. Three problems distributed among these tests cover student understanding of the need to engage in continuing professional development for any professional working in the field of computing. These questions are structured as case studies that present a real world situation where students are asked to suggest possible solutions involving the analysis of one or more possible scenarios and/or outcomes. Measurable Performance Criteria: Student grades on the three problems are averaged and converted to the 1 to 4 point scale. That score is used to assess the Student Outcome (h) as follows: Not Met (1.00-1.99), Minimally Attained (2.00-2.79), Met (2.80-3.59), Exceeded (3.60-4.00).

5.2.8.2 Special scoring rubric in CS 498 – Senior Project

Assessment Instrument: Students in CS 498 are required to complete a substantial course project serving as a capstone in this undergraduate program. As a part of the course project requirements, students are expected to identify and become proficient with one or more new programming language, development environment, modeling tool, etc., with which they were not familiar previously, and actively use it throughout the project. The professional development aspect of the course project is graded using two rubrics. The instructor completes one rubric, while the students complete the other rubric to assess themselves. Each rubric includes such criteria as recognizing the need for professional development, choice of development area, applicability, mastery, and usefulness for future projects. Measurable Performance Criteria: Each criterion on the teamwork assessment rubrics is graded on the 4-point scale where Poor = 0, Fair = 1, Average = 2, Good = 3, Excellent = 4. The score on the scale of 1 to 4 for each rubric is calculated by averaging the grades on each criterion on the respective rubric. Assessment score is calculated using the as follows:

\[ \text{Assessment score (1 to 4) = (Instructor score) \times 0.7 + (Student score) \times 0.3} \]
5.2.9  **SO i: An ability to use current techniques, skills, and tools necessary for computing practice**

Student Outcome (i) is assessed using the following instruments:

5.2.9.1  **Course project assessment rubric in CS 410 – Software Engineering**

*Assessment Instrument:* Students in CS 410 are required to complete a substantial course project. While working on the project, students need to use a broad range of current techniques, skills, and tools necessary for the successful completion of the course project. Students are expected to have the proper skills to be proficient with such tools as an integrated development environment (IDE) for a modern programming language, a UML modeling tool, and a version control system. The skills and tools aspect of the course project is graded using an assessment rubric.

*Measurable Performance Criteria:* Each skill and tool criterion on the assessment rubric is graded on the 4-point scale where Poor = 0, Fair = 1, Average = 2, Good = 3, Excellent = 4. The score on the scale of 1 to 4 for each rubric is calculated by averaging the grades on each criterion.

5.2.9.2  **Course project assessment rubric in CS 498 – Senior Project**

*Assessment Instrument:* Students in CS 498 are required to complete a substantial course project. While working on the project, students need to use a broad range of current techniques, skills, and tools necessary for the successful completion of the course project. Students are expected to have the proper skills to be proficient with such tools as an integrated development environment (IDE) for a modern programming language, a UML modeling tool, and a version control system. The skills and tools aspect of the course project is graded using an assessment rubric.

*Measurable Performance Criteria:* Each skill and tool criterion on the assessment rubric is graded on the 4-point scale where Poor = 0, Fair = 1, Average = 2, Good = 3, Excellent = 4. The score on the scale of 1 to 4 for each rubric is calculated by averaging the grades on each criterion.
5.2.10 **SO 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**

Student Outcome (j) is assessed using the following instruments:

5.2.10.1 **Embedded assessment in student exams in CS 385 – Computer Architecture**

Assessment Instrument: Students taking the Computer Architecture course learn fundamental concepts of Computer Science as instruction set architectures, computer arithmetic, memory systems and parallel processing, and the way these concepts are applied to develop efficient computing systems based on pipelining and multiprocessing. They also learn a modern industry-standard tool for designing computer systems, the Hardware Description Language (Verilog HDL). The students are required to take a midterm test and a final exam, which measure their level of mastering of these concepts, techniques and tools. The tests include comprehensive sets of questions covering the related course learning outcomes and are used to measure the corresponding student outcome.

Measurable Performance Criteria: Each of the midterm test and the final exam consists of 20 questions altogether covering the Computers Architecture concepts and techniques learnt in the course. The student answer to each question is graded on a 0.0-0.1 point scale. The sum of the scores from all 40 questions is then used to assess the Student Outcome j (SO j) as follows: Not Met (1.00-1.99), Minimally Attained (2.00-2.79), Met (2.80-3.59), Exceeded (3.60-4.00).

5.2.10.2 **Embedded assessment in student exams in CS 354 – Digital Systems Design**

Assessment Instrument: The Digital Design course includes a theoretical material on Boolean algebra and finite state machines, which are part of the fundamental Computer Science theory. The students learn specific techniques for analyzing and designing Boolean functions and finite state machines and mapping them into digital circuits as logic gates and memory elements. For this purpose various minimization techniques are used and students learn how to apply them and make design choices in order to create optimal solutions. The students are required to take three tests including the final exam, which measure their level of mastering the theories, concepts and techniques they learn in the Digital Design course. The final exam includes a comprehensive set of questions on the theory and problems on analysis and design of digital circuits and is used to measure the student outcome for these course learning outcomes.

Measurable Performance Criteria: The final exam consists of 10 questions and the student answer to each one is graded on a 100% scale. The average of the student scores from all questions is then converted into a 4-point system and used to assess the Student Outcome j (SO j) as follows: Not Met (1.00-1.99), Minimally Attained (2.00-2.79), Met (2.80-3.59), Exceeded (3.60-4.00).

5.2.10.3 **Assessment rubric for the student course project in CS 355 – Systems Programming**

Assessment instrument: All students in the Systems Programming course (CS 355) are required to complete a team-based course project. The objective of the project is to design and develop a complex software system that meets a set of specified requirements with the features ranging from providing a high-level user interface, meeting mid-level data storage considerations, to ensuring a correct low-level interaction with a hardware controller. Students are required to make design choices to balance the complexity of the system by distributing its features across various levels of architecture granularity and abstraction paradigms. The project is assessed using a specially developed rubric.

Measurable performance criteria: Instructor evaluations used an assessment rubric that included assessment criteria for four levels of performance: Not Met (1.00-1.99), Minimally Attained (2.00-2.79), Met (2.80-3.59), and Exceeded (3.60-4.00). The assessment score is calculated as the average student score.

5.2.10.4 **Major Field Test indicator “Discrete Structures and Algorithms”**

Assessment instrument: All graduating seniors are required to complete the Major Field Test (MFT) in the Spring of their final year to test their knowledge across a comprehensive set of topics covered by the entire program. The test consists of 66 questions and lasts 120 minutes. MFT results are comprised of three assessment indicators, one of which is titled “Discrete Structures and Algorithms.”

Measurable performance criteria: The objective level of difficulty of this test varies from year to year; however, all students taking the test nationwide receive the same set of questions. Therefore, the
percentile of correct answers for the student group taking the test during a given semester is used to assess student performance on this test with the following qualitative metrics: 0% to 24% = Not Met (1.00-1.99), 25% to 44% = Minimally Attained (2.00-2.79), 45% to 64% = Met (2.80-3.59), and 65% to 100% = Exceeded (3.60-4.00). The assessment score is calculated as:

\[
\text{Assessment score (1 to 4) } = \frac{\text{percentile}}{25} + 1.0
\]
5.2.11 SO k: An ability to apply design and development principles in the construction of software systems of varying complexity

Student Outcome (k) is assessed using the following instruments:

5.2.11.1 Course project assessment rubric in CS 410 – Software Engineering

Assessment Instrument: Students in CS 410 are required to complete a substantial course project. In this project, students need to apply design and development principles in the construction of two-tiered web-based software system. The design and development aspect of the course project is graded using an assessment rubric, which includes criteria covering the categories related to system requirements, system architecture design, project planning, as well as unit, functional, integration, and system testing.

Measurable Performance Criteria: Each design and development criterion on the assessment rubric is graded on the 4-point scale where Poor = 0, Fair = 1, Average = 2, Good = 3, Excellent = 4. The score on the scale of 1 to 4 for each rubric is calculated by averaging the grades on each criterion.

5.2.11.2 Course project assessment rubric in CS 498 – Senior Project

Assessment Instrument: Students in CS 498 are required to complete a substantial course project. In this project, students need to apply design and development principles in the construction of two-tiered web-based software system. The design and development aspect of the course project is graded using an assessment rubric, which includes criteria covering the categories related to system requirements, system architecture design, project planning, as well as unit, functional, integration, and system testing.

Measurable Performance Criteria: Each design and development criterion on the assessment rubric is graded on the 4-point scale where Poor = 0, Fair = 1, Average = 2, Good = 3, Excellent = 4. The score on the scale of 1 to 4 for each rubric is calculated by averaging the grades on each criterion.

5.2.11.3 Major Field Test indicator “Systems: Architecture, Operating Systems, Networking, Database”

Assessment instrument: All graduating seniors are required to complete the Major Field Test (MFT) in the Spring of their final year to test their knowledge across a comprehensive set of topics covered by the entire program. The test consists of 66 questions and lasts 120 minutes. MFT results are comprised of three assessment indicators, one of which is titled “Systems: Architecture, Operating Systems, Networking, Database.”

Measurable performance criteria: The objective level of difficulty of this test varies from year to year; however, all students taking the test nationwide receive the same set of questions. Therefore, the percentile of correct answers for the student group taking the test during a given semester is used to assess student performance on this test with the following qualitative metrics: 0% to 24% = Not Met (1.00-1.99), 25% to 44% = Minimally Attained (2.00-2.79), 45% to 64% = Met (2.80-3.59), and 65% to 100% = Exceeded (3.60-4.00). The assessment score is calculated as:

Assessment score (1 to 4) = percentile / 25 + 1.0
5.3. Course-level Assessment
Assessment of each course is completed using the current Course Assessment Report Template, which outlines the common standards for assessing courses offered in the program. Instruments used to assess SO’s outlined in Section 4 are embedded in the assessment of the corresponding individual courses. Assessment of each course is typically completed within a month (excluding the winter and summer breaks) after that course is offered.
6. Evaluation

As defined in the current ABET criteria [1],

*Evaluation is one or more processes 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.*

6.1. Attainment of Student Outcomes

Using the current SO assessment data and the corresponding criteria outlined in Section 5.2, SO’s are interpreted to determine whether each SO is exceeded, met, minimally attained, or not met. The department meets to make these evaluation decisions during the first month of the semester (i.e. in January to evaluate the results from the previous Fall semester, and in September to evaluate the results from the previous Spring semester).

For each SO with updated assessment data, the following process takes place:

- If all assessment metrics indicate that the given SO *meets* or *exceeds* the criteria, no action is required;
- If at least one assessment metric indicates that a given SO is *minimally attained*, the course adjustment procedure (Section 6.3) is initiated for the course(s) used in the assessment of this SO whose assessment indicates a deficiency;
- If at least one assessment metric indicates that a given SO *does not meet* the criteria, a targeted subset of courses that support this SO (Table 2) is selected, and a course adjustment procedure (Section 6.3) is initiated for these courses. This subset will include all courses used to assess this SO whose assessment indicates a deficiency;
- If one of the assessment metrics contributed by the MFT is either minimally attained or does not meet the criteria, targeted subset of courses that support the corresponding SO (Table 2) is selected, and a course adjustment procedure (Section 6.3) is initiated for these courses. If necessary, a discussion will be initiated whether the required courses are sufficient to provide the required coverage of a SO, or whether additional coverage of that SO should be added to another existing required course.

Trends for each SO over several semesters are observed. If there is an observable decline in the levels of student attainment of a particular SO, the coverage of that SO will be reviewed to determine whether it is adequately covered in the required courses.

6.2. Course-level Evaluation

Using the same schedule as outlined in Section 6.1, the current “Course Assessment Report Template” is used by the course coordinators evaluate whether each Course Learning Outcomes (CLO) meet the corresponding criteria.

For each course taught in the previous semester, a brief discussion of the assessment and evaluation results takes place at a department meeting. For any courses with at least one CLO below the target criteria, a course adjustment procedure (Section 6.3) is initiated.

6.3. Course Adjustments as a Result of Evaluation

Course content, structure, and assessment methods may need to be adjusted if the evaluation of the previous assessment data for that course indicates a deficiency in the student attainment of a CLO and/or a corresponding SO. The aim of such a course adjustment is to improve student success as measured by a CLO and/or an SO attainment metrics described in Section 5.2.

When a deficiency is identified, the course content and structure may need to be adjusted to strengthen the coverage of the course material relevant to improving the student mastery of the currently deficient CLO and/or SO. For each CLO that is below the target criteria, course coordinator (in consultation with other faculty members) thoroughly reviews the course material to identify and suggest modifications that would strengthen student success in the areas that need improvement. Additionally, assessment instruments may also need to be adjusted to better reflect the level of student performance in the deficient areas.
Any course adjustment will be deemed successful if the subsequent assessment data and its evaluation indicate that the previously identified deficiencies are resolved and the corresponding CLO(s) and/or SO(s) is/are being met.

Any modifications detailed in this section will be implemented during the next course offering as described in the following schedule:

- A course is offered in semester X;
- Assessment and evaluation is completed during the first month of semester X+1 and deficiencies are identified;
- Modifications are identified by mid point of semester X+1 and are scheduled for implementation during semester X+2;
- Course is offered again (with modifications) in semester X+2;
- Assessment and evaluation is completed the first month of semester X+3, and a decision is made about the success of the course adjustment process by comparing the current assessment results with those from the previous course offering.

7. References


Document change log

<table>
<thead>
<tr>
<th>Rev.</th>
<th>Date</th>
<th>Reason for change</th>
<th>Author</th>
</tr>
</thead>
<tbody>
<tr>
<td>11/21/2013</td>
<td>First draft</td>
<td></td>
<td>Kurkovsky</td>
</tr>
<tr>
<td>12/05/2013</td>
<td>CS 253: &quot;A&quot; moved from SO b to SO c</td>
<td>Kurkovsky</td>
<td></td>
</tr>
<tr>
<td>12/16/2013</td>
<td>Added info on SO assessment (Section 5.2) for CS 151, 253, 354, 385</td>
<td>Kurkovsky</td>
<td></td>
</tr>
<tr>
<td>1/2/2014</td>
<td>Added remaining SO assessment materials and finalized Section 6.1</td>
<td>Kurkovsky</td>
<td></td>
</tr>
</tbody>
</table>