L&S ASSESSMENT REPORT SURVEY 2012-2013

Recent Assessment Activity

Q5.1.Assessment Purpose. Please describe the purpose of the assessment activity (e.g., to conduct a curricular or program review, to assess learning across a sequence of courses, to solve a problem with student performance, to honor MIU or other obligations, etc.)

The purpose of this assessment is to solve a problem with student performance.

Q5.2. Learning Outcomes or Goals Assessed. Referring to the list of student learning objectives/goals expressed in the program assessment plan, please identify the learning outcome(s) that were the focus of the assessment activity: what did you study about what you want students to know, value and/or do?

Two of the learning objectives for CS undergraduates are:

1. Students can create robust, user-friendly, well-structured and well-documented programs in current programming languages such as C.
2. Students demonstrate a basic understanding of how modern computers work in both hardware (e.g., computer architecture) and software (e.g., operating systems, compilers, or networking)

These two specific learning objectives are expected to be partially addressed in CS 354. Focusing in more detail, we have the following goals for students in CS 354:

1. Students can read and understand complex sequences of x86 assembly code
2. Students can read, understand, and generate serious projects in the C programming language
3. Students can use the related gcc toolchain, including compiler, debugger, and disassembler
4. Students can use a real code editor (e.g., emacs or vim)

(Identified as Issue A in Assessment Plan from Spring 2013.)

Q5.3. Assessment Strategy. Please describe your most recent assessment project. What did you try to better understand student learning across this program, in the context of the learning goal discussed in your response above? Feel free to describe the tools, strategies, methods, and analysis used (e.g., graduating student surveys, standardized tests, grades on embedded questions on exams, alumni surveys, focus groups or interviews, evaluation of student work on papers, portfolios, capstone assignments, etc.)

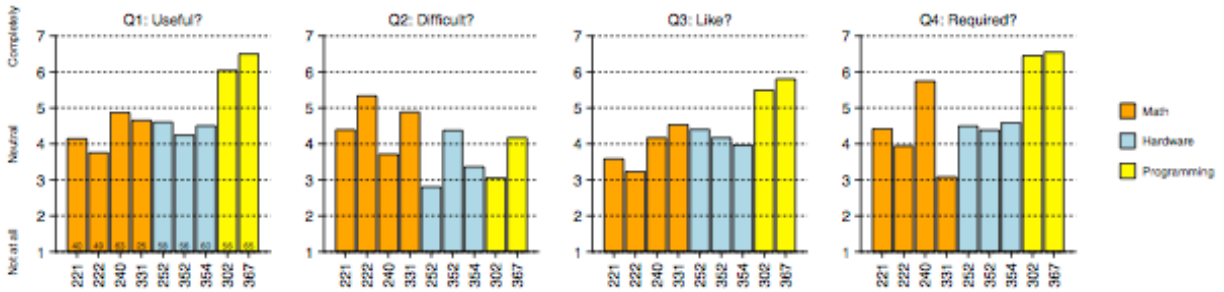
Instructors in some upper-level courses (e.g., CS 537, CS 536, CS 564) were finding that undergraduates were not adequately prepared for the intense amount of programming required. In particular, while students could program well in Java (taught in CS 302 and 367) they were unprepared for developing larger programs and for programming and debugging in languages such as C or C++.

To better understand some of the issues with our lower-level courses, in 2010 we surveyed upper-level students in three courses covering a broad range of CS specialties: CS 520 (Theory), 537 (OS), and 540 (AI). A total of 69 students were queried; answers were removed from non-majors (thus only CS and ECE student replies were counted). Four questions were asked:

- Is the course useful to you? (1–7)
- Was the material difficult? (1–7)
- Did you like the course? (1–7)

- Should it be required for the major? (1–7)

Each question was to be answered on a scale of 1 to 7, where 1 means “not at all”, 4 means “neutral”, and 7 means “completely.” The graphs below present the results.



Q5.4. Key Findings and Impact. Please summarize the key findings (evidence/results) and how the department or program plans to use this information (e.g., no curricular changes, program enhancements, program redesign, etc.). This may include to whom results were reported to effect change (if needed),whether the results suggested other areas of inquiry,plans for continued attention to assessment (including "tweaking" the assessment plan),and/or deadlines for achieving milestones related to the above activities.

From the graphs, a number of points stand out. First, the programming sequence (including 302 and 367) seems to be most universally liked and thought useful. However, CS 354 is viewed as only reasonably useful, not very difficult, somewhat liked, and not as strongly necessary for the major; this data about CS 354 in conjunction with the observations from upper-level instructors, led the department to begin addressing these problems.

To begin, in Fall 2012 Professor Remzi Arpaci-Dusseau taught an experimental section of CS 354 to update the course and focus more on systems programming in C. Five new intense programming assignments were adopted.

Students were then surveyed for their opinions on these projects:

- Difficulty: Too easy (1), Just right (2), Too hard (3)
- Interest: Boring (1), OK (2), Interesting (3)
- Value: Learned little (1), Learned some (2), Learned lots (3)

The averages for the projects were:

Project	Difficulty	Interest	Value
Intro to C	1.9 (just right)	2.2 (OK+)	2.4 (learned some/lots)
Binary Bombs	1.6 (easy/just right)	2.3 (OK+)	2.3 (learned some/lots)
Malloc/Free	2.3 (just right/hard)	2.5 (OK/interesting)	2.6 (learned some/lots)
Cache Simulator	2.1 (just right)	2.4 (OK/interesting)	2.5 (learned some/lots)
Web Server/Proxy	2.2 (just right+)	2.7 (interesting)	2.8 (learned lots)

From these results, we infer that most students found the projects to be just about the right level of difficulty; most students also found the projects fairly interesting (especially the more complex, more involved projects with more C-code needing to be written); finally, most students learned a lot all the projects. Overall feedback from students about the redesigned course was extremely positive.

As a result, and in consultation with the systems faculty, we plan to incorporate these changes into all of the sections of CS 354, hopefully starting in Fall 2013.

The Future

Q6.1. Please let us know what your next steps for assessing student learning will be.

We will evaluate whether or not the more intense programming projects scale to the large number of students enrolled in all sections CS 354.

Thank you!

Recent Assessment Activity

Q5.1.Assessment Purpose. Please describe the purpose of the assessment activity (e.g., to conduct a curricular or program review, to assess learning across a sequence of courses, to solve a problem with student performance, to honor MIU or other obligations, etc.)

The purpose of the assessment was to conduct a curricular review.

Q5.2. Learning Outcomes or Goals Assessed. Referring to the list of student learning objectives/goals expressed in the program assessment plan, please identify the learning outcome(s) that were the focus of the assessment activity: what did you study about what you want students to know, value and/or do?

We wanted to assess our high-level goal of ensuring that both our undergraduate and graduate students are well prepared for a successful CS career.

(Identified as Issue B in Assessment Plan from Spring 2013.)

Q5.3. Assessment Strategy. Please describe your most recent assessment project. What did you to to try to better understand student learning across this program, in the context of the learning goal discussed in your response above? Feel free to describe the tools, strategies, methods, and analysis used (e.g., graduating student surveys, standardized tests, grades on embedded questions on exams, alumni surveys, focus groups or interviews, evaluation of student work on papers, portfolios, capstone assignments, etc.)

During our May 2013 Departmental Lunch, we surveyed all attending undergraduate CS majors and graduate students. For this assessment, there are three relevant questions, all are on a scale of 1 (Not at all) to 7 (Definitely):

1. When you graduate, how likely are you **to choose** to work in a position related to Computer Science?
2. When you graduate, how likely do you think you will be able **to find** a job related to Computer Science?
3. When you graduate, how **prepared** will you feel to work in a position related to Computer Science?

Of the 55 undergraduate CS majors who responded, the means were as follows: 6.4, 6.3, and 5.8. While most of the students are very likely to choose a career related to CS and are highly confident they will be able to find a CS job, a more noticeable number of undergraduates feel (slightly) under-prepared for a CS career.

For comparison, we note that the graduate CS students (both M.S. and Ph.D.) responded slightly higher: 6.8, 6.6, and 6.6. Although this was a very small sample of only 14 graduate students (the lunch was primarily for undergraduates), we did not see further issues to investigate for graduate students at this time.

To understand these results for undergraduates in more detail, we examined the responses to another survey question:

1. Are there courses or topics that you wish were offered in CS but are not?

We provided no suggestions or leading questions and the responses were completely free form. Of the 55 undergraduate respondents, 30 students (nearly 55%) specified that they wish additional CS courses were offered; all but 1 of those respondents asked for courses focusing on the practical coding-intensive aspects of computing (e.g., security, data mining, software engineering, parallel computation, and other issues related to system programming). A high number of students (12 out of 55) specifically requested a course related to web development and programming.

Q5.4. Key Findings and Impact. Please summarize the key findings (evidence/results) and how the department or program plans to use this information (e.g., no curricular changes, program enhancements, program redesign, etc.). This may include to whom results were reported to effect change (if needed),whether the results suggested other areas of inquiry,plans for continued attention to assessment (including "tweaking" the assessment plan),and/or deadlines for achieving milestones related to the above activities.

While the curriculum committee and the department has not yet had a chance to reflect on these results, we will need to evaluate the demand for additional "practical" coding-intensive CS courses. These new courses may be similar to project-heavy ones that the department has recently approved: Software Engineering (CS 506 and 706) and Mobile Applications (CS 407). We may also consider ways to find the resources to cover CS 369: Web Programming which we have not been able to offer in many years.

Another option is to consider introducing more special topics courses on current programming languages and environments. To fulfill this role, the department already has in place a 1-credit course called CS 368: Learning a New Programming Language. Currently, we are able to offer two sections: one covering C++ and one covering Matlab, both of which are in high demand. We may investigate offering additional sections on other current topics.

Recent Assessment Activity

Q5.1.Assessment Purpose. Please describe the purpose of the assessment activity (e.g., to conduct a curricular or program review, to assess learning across a sequence of courses, to solve a problem with student performance, to honor MIU or other obligations, etc.)

The purpose of the assessment was to conduct a curricular review.

Q5.2. Learning Outcomes or Goals Assessed. Referring to the list of student learning objectives/goals expressed in the program assessment plan, please identify the learning outcome(s) that were the focus of the assessment activity: what did you study about what you want students to know, value and/or do?

We wanted to begin to assess whether or not all CS students should be required to complete the three introductory courses of CS 302, CS 367, and CS 240 and obtain a minimum GPA before they can declare a CS major. We had two preliminary questions to answer. First, do students perceive the utility of these three courses? Second, does student performance on these three courses (specifically, final letter grades) predict performance in the CS major (specifically, GPA in the major at graduation disregarding those three courses)?

(Identified as Issue C in Assessment Plan from Spring 2013.)

Q5.3. Assessment Strategy. Please describe your most recent assessment project. What did you to to try to better understand student learning across this program, in the context of the learning goal discussed in your response above? Feel free to describe the tools, strategies, methods, and analysis used (e.g., graduating student surveys, standardized tests, grades on embedded questions on exams, alumni surveys, focus groups or interviews, evaluation of student work on papers, portfolios, capstone assignments, etc.)

During our Spring 2013 Departmental Lunch, we surveyed the 55 undergraduate students about their perception of the need for CS degree requirements. Specifically, we asked these two related questions:

1. CS 302, CS 367, and CS 240 are all required to declare a CS major. Are there any of those courses you would NOT have taken if not required? If so, why?

2. To complete your degree, you must take:
   - one course in Theory (CS 520 or CS 577),
   - two courses in Hardware/Software (CS 407, CS 536, CS 537, CS 538, CS 552, CS 564, CS 640, or CS 642)
   - one course in Applications (CS 412, 416, 425, 513, 514, 515, 525, 534, 540, 545, 547, 559, or 570).

   Please circle the four courses you anticipate taking to fulfill our requirements. Are there any of those four courses you would NOT take, if not required? If so, why?

To assess our second question of whether student performance on 302, 367 and 240 (specifically, final letter grades) predict performance in the CS major (specifically, GPA in the major at graduation disregarding those three courses), we examined the official grades assigned on each CS course to each CS major who graduated in the last 3 years.

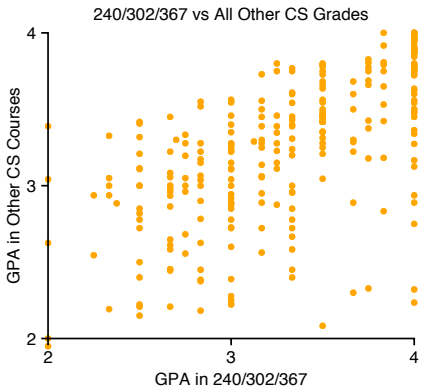
Q5.4. Key Findings and Impact. Please summarize the key findings (evidence/results) and how the department or program plans to use this information (e.g., no curricular changes, program enhancements, program redesign, etc.). This may include to whom results were reported to effect change (if needed),whether the results suggested other areas of inquiry,plans for continued attention to assessment (including "tweaking" the assessment plan),and/or deadlines for achieving milestones related to the above activities.

For question 1, 27 out of 55 students (nearly 50%) commented that they would not have taken CS 240 if it was not required or that they had some negative experience with the course. In contrast, only one student commented about CS 302 or CS 367, saying that CS 302 was too easy and therefore shouldn't be required.

Again, in contrast, for question 2, relatively few students stated that they would not have taken one of the required upper-level courses. Specifically, only four students would not have taken theory, four would not have taken applications, and two would not have taken one of the two required courses in systems.

While student perception is not our only guide for determining whether or not a fundamental course is needed as a prerequisite, the negative perceptions do indicate we must investigate CS 240 further. However, the curriculum committee and faculty have not yet had an opportunity to react to these findings.

The results of our second question concerning the correlation between introductory grades and CS major GPA are shown below.



The results indicate that there is a positive correlation between GPA in 240, 302, and 367 and later CS grades with a Pearson coefficient around 0.60. This results indicates that grades in these courses may be able to be used as a weak filter for selecting students who will perform well as CS majors.

The Future

Q6.1. Please let us know what your next steps for assessing student learning will be.

This data will be used by the CS Undergraduate Advising Committee, the Curriculum Committee, and the faculty to guide a discussion on whether these three courses and their GPA should be required to declare a CS major.

Thank you!