Computer Science Undergraduate Program Assessment Process

A mapping between learning outcomes and objectives for each course grounds the Assessment Process within the Computer Science Department. The Computer Science undergraduate program learning objectives can be found at: https://www.pdx.edu/degmap/sites/www.pdx.edu.degmap/files/useful_links/PLLO%20-%20%20CS.pdf and are repeated here for convenience:

1. Adapt algorithms and data structures drawn from a large standard repertoire to new problems.
2. Assess new developments in computer science.
3. Communicate with other members of development teams and with customers.
4. Computing at all levels of abstraction, including: (a) circuits and computer architecture; (b) operating systems; (c) programming languages, and (d) algorithms.
5. Debug and test programs.
6. Develop program designs from specifications under a variety of software paradigms/ architecture.
7. Develop program specifications from a variety of informal descriptions.
8. Engineering principles used to meet the challenge of large-scale software production.
9. Implement selected designs as programs in a variety of programming languages.
10. Mathematical foundations of computer science.
11. Perform quantitative evaluation of program behavior by experiment.
12. Present the results of their work as written technical documents.
13. Present the results of their work orally.
14. The ethical and legal responsibilities of computing professionals.
15. The impact of computing on society.
16. The interdependence of hardware and software.
17. The management and sharing of persistent data.
18. Use analytical techniques to evaluate and compare different designs that meet specifications.

The faculty will typically review the program outcomes once each year, as part of the annual faculty retreat. A new program mapping will be required whenever a new set of program outcomes is adopted.

The department maintains a catalog of the courses that it offers at https://www.pdx.edu/computer-science/undergraduate-courses. The catalog may be updated periodically as new courses are introduced, as old courses are retired, or as new versions of courses are introduced as replacements for old. For each course in the catalog, there is a set of course objectives that itemizes specific skills or knowledge that students are expected to have "upon successful completion" of that course. Every instructor who teaches the course should be familiar with the list of course objectives, and should share it with the students early in the course to ensure that appropriate expectations are set.

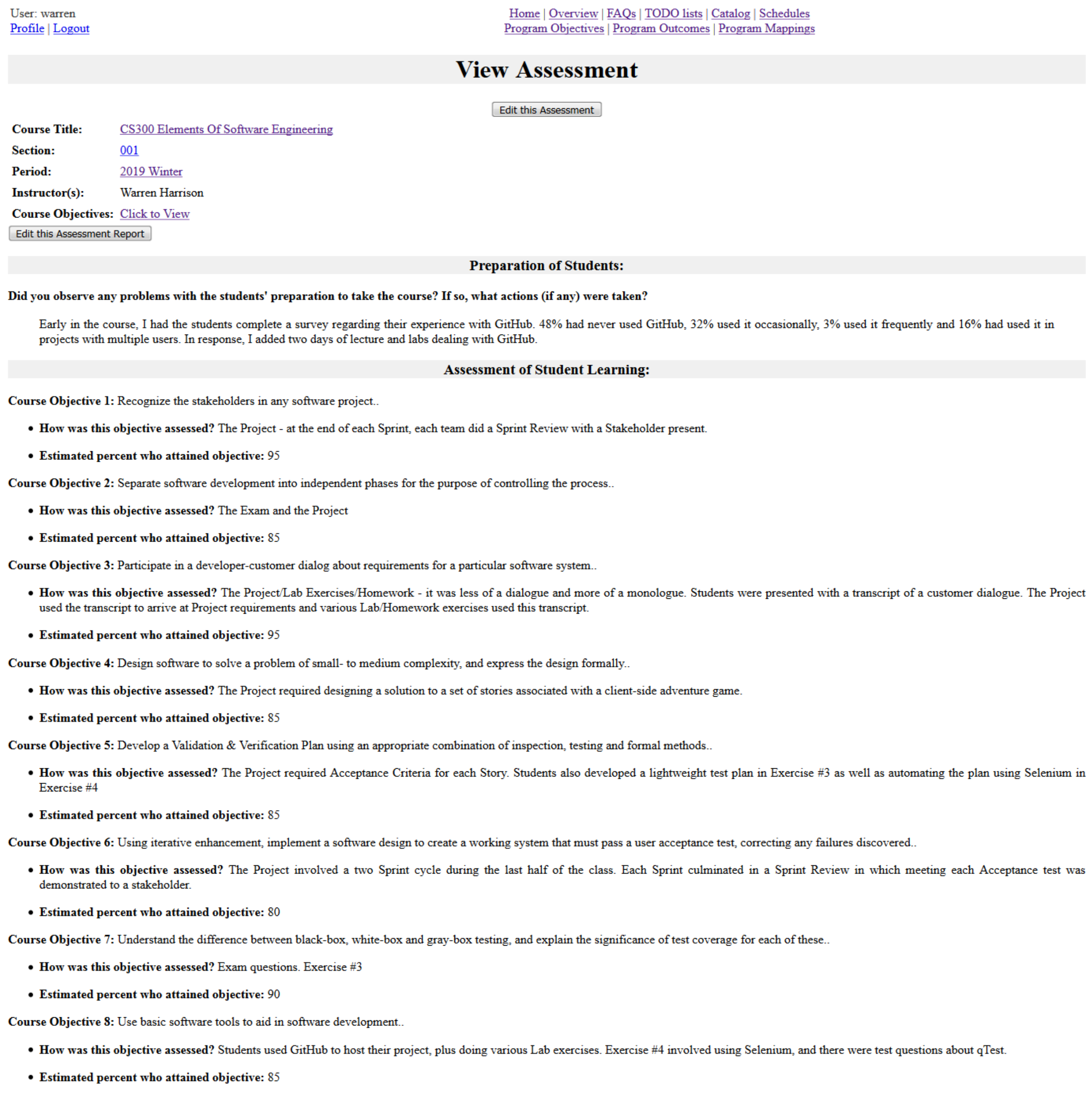
Associated with each set of course objectives, is a mapping that describes how individual course objectives relate to program level learning outcomes. The mappings can be used to help determine whether program learning outcomes are being sufficiently covered by the material that is taught in individual, required courses.

For instance, this is a mapping from the course objectives for CS300 to the program learning outcomes (numbers refer to the numbered learning objectives on the previous page):

Course Objectives/Learning Objectives	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18
1. Recognize the stakeholders in any software project.			X											X	X			
2. Separate software development into independent phases for the purpose of controlling the process.			X			X	X	X				X	X			X		X
3. Participate in a developer-customer dialog about requirements for a particular software system.			X			X	X					X	X	X				
4. Design software to solve a problem of small- to medium complexity, and express the design formally.	X					X	X					X	X					
5. Develop a Validation & Verification Plan using an appropriate combination of inspection, testing and formal methods.			X		X									X	X			
6. Using iterative enhancement, implement a software design to create a working system that must pass a user acceptance test, correcting any failures discovered.	X		X		X	X	X	X	X	X								
7. Understand the difference between black-box, white-box and gray-box testing, and explain the significance of test coverage for each of these.					X									X				
8. Use basic software tools to aid in software development.		X	X															
Summary	X	X	X		X	X	X	X	X	X		X	X	X	X	X		X

Regardless of other differences, the instructor in each section is expected to cover all of the objectives that have been defined for that course. Conversely, if a topic is not mentioned in the list of objectives then it is possible that an instructor for that course may not include coverage of that topic. This mechanism is intended to provide some significant degree of consistency, at least with respect to technical content, between different offerings of the same course.

Prior to the start of a course, instructors are expected to review the course objectives, and at the end of each term, instructors are expected to complete an online assessment report to document coverage of course objectives and to reflect on possible future revisions:



User: warren
Profile | Logout

Home | Overview | FAQs | TODO lists | Catalog | Schedules
Program Objectives | Program Outcomes | Program Mappings

View Assessment

Edit this Assessment

Course Title:	CS300 Elements Of Software Engineering
Section:	001
Period:	2019 Winter
Instructor(s):	Warren Harrison
Course Objectives:	Click to View

Edit this Assessment Report

Preparation of Students:

Did you observe any problems with the students' preparation to take the course? If so, what actions (if any) were taken?

Early in the course, I had the students complete a survey regarding their experience with GitHub. 48% had never used GitHub, 32% used it occasionally, 3% used it frequently and 16% had used it in projects with multiple users. In response, I added two days of lecture and labs dealing with GitHub.

Assessment of Student Learning:

Course Objective 1: Recognize the stakeholders in any software project..

- **How was this objective assessed?** The Project - at the end of each Sprint, each team did a Sprint Review with a Stakeholder present.
- **Estimated percent who attained objective:** 95

Course Objective 2: Separate software development into independent phases for the purpose of controlling the process..

- **How was this objective assessed?** The Exam and the Project
- **Estimated percent who attained objective:** 85

Course Objective 3: Participate in a developer-customer dialog about requirements for a particular software system..

- **How was this objective assessed?** The Project/Lab Exercises/Homework - it was less of a dialogue and more of a monologue. Students were presented with a transcript of a customer dialogue. The Project used the transcript to arrive at Project requirements and various Lab/Homework exercises used this transcript.
- **Estimated percent who attained objective:** 95

Course Objective 4: Design software to solve a problem of small- to medium complexity, and express the design formally..

- **How was this objective assessed?** The Project required designing a solution to a set of stories associated with a client-side adventure game.
- **Estimated percent who attained objective:** 85

Course Objective 5: Develop a Validation & Verification Plan using an appropriate combination of inspection, testing and formal methods..

- **How was this objective assessed?** The Project required Acceptance Criteria for each Story. Students also developed a lightweight test plan in Exercise #3 as well as automating the plan using Selenium in Exercise #4
- **Estimated percent who attained objective:** 85

Course Objective 6: Using iterative enhancement, implement a software design to create a working system that must pass a user acceptance test, correcting any failures discovered..

- **How was this objective assessed?** The Project involved a two Sprint cycle during the last half of the class. Each Sprint culminated in a Sprint Review in which meeting each Acceptance test was demonstrated to a stakeholder.
- **Estimated percent who attained objective:** 80

Course Objective 7: Understand the difference between black-box, white-box and gray-box testing, and explain the significance of test coverage for each of these..

- **How was this objective assessed?** Exam questions. Exercise #3
- **Estimated percent who attained objective:** 90

Course Objective 8: Use basic software tools to aid in software development..

- **How was this objective assessed?** Students used GitHub to host their project, plus doing various Lab exercises. Exercise #4 involved using Selenium, and there were test questions about qTest.
- **Estimated percent who attained objective:** 85

Each course has a course coordinator. A course coordinator's role is to ensure that the course description, including the set of objectives, is appropriate to the level, subject, and length of the course. Course coordinators also maintain the course description with the help of feedback from each instructor who teaches the course through the assessment reports that they complete at the end of the term. This information is collated and presented to the coordinator at the annual course coordinator meeting that is intended for all faculty, and is currently held in the Spring term.

Program Assessment Status for 2018-2019

Program learning objectives are met through the collection of required CS courses (CS core) that are taken by every CS major. CS Electives also have objectives, many of which address the program outcomes, but each student could take a different combination of CS electives, so the only classes students are guaranteed to have taken is the CS core.

The CS core, and their online assessment report status for the 2018-2019 academic year:

Core CS Course	**FALL**	**WTR**	**SPR**
CS162 Introduction to Computer Science	YES	YES	IP
CS163 Data Structures	YES	YES	IP
CS201 Computer Systems Programming	YES	YES	IP
CS202 Programming Systems	YES	YES	IP
CS250 Discrete Structures I	YES	YES	IP
CS251 Discrete Structures II	NO	YES	IP
CS300 Elements Of Software Engineering	YES	YES	IP
CS305 Social, Ethical, and Legal Implications of Computing	NO	YES	IP
CS320 Principles of Programming Languages	YES	YES	IP
CS333 Introduction to Operating Systems	YES	YES	IP
CS350 Algorithms and Complexity	YES	YES	IP
CS469/470 Software Engineering Capstone I/II (assessed as a pair at the completion of CS470)	YES	NO	IP