1. ASSESSMENT PROGRAM CRITERIA.

In order to properly fulfill our mandate to assess and improve the quality of undergraduate education in Computer Science at UGA, our assessment program:

1. Must facilitate the genuine commitment by the CS faculty to improve the quality of undergraduate education;
2. Must be based on a clearly defined model of what variables are be measured and how;
3. May involve qualitative measures, but must involve quantitative measures;
4. Must use multiple independent measures for each variable to be measured;
5. Must be linked into a formal feedback mechanism for improving undergraduate instruction based on the outcomes of the assessment.

We understand that to lessen the burden on departments, the Administration has moved from annual assessment reports to a three-year cycle, but that they expect a correspondingly more rigorous report. In order to react effectively to the needs of our fast-changing discipline, our assessment program will continue to create internal reports and recommendations annually or more frequently.

To minimize the administrative burden on our faculty arising from this program, and also to make the results more intelligible, our model focuses on a small number of key outcomes that measure overall program success. Our success in these key outcomes will be estimated via measurements of specific course instructional goals.
2. PROPOSED CS ASSESSMENT MODEL

This model is adapted from the model used by Biological and Agricultural Engineering, who submit detailed annual reports as mandated by their professional accreditation process. Ours is a less involved version.

A. Undergraduate program outcomes: We base our desired outcomes on the Computer Science undergraduate curriculum guidelines proposed by the Association for Computing Machinery, the primary professional organization for our discipline\(^1\). From these guidelines we have extracted the following "Modal standard," representing the average level of achievement expected of CS graduates. The modal standard consists of seven objectives:

1. Demonstrate a sound understanding of the main areas of the body of knowledge and the theories of computer science, with an ability to exercise critical judgment across a range of issues.

2. Critically analyze and apply a range of concepts, principles, and practices of the subject in the context of loosely specified problems, showing effective judgment in the selection and use of tools and techniques.

3. Produce work involving problem identification, analysis, design, and development of a system, along with appropriate documentation. The work must show a range of problem solving and evaluation skills, draw upon supporting evidence, and demonstrate a good understanding of the need for quality.

4. Demonstrate the ability to work as an individual with minimum guidance and as either a leader or member of a team.

5. Follow appropriate practices within a professional, legal, and ethical framework.

6. Identify mechanisms for continuing professional development and life-long learning.

7. Explain a wide range of applications based upon the body of knowledge.

\(^1\) ACM Curriculum Committee 2001 recommendations, Steelman draft
B. Specific "landmark" course objectives: As students go through their undergraduate program, they all pass through the following "landmark" courses:

- The initial computer programming courses, CSCI 1301 and 1302;
- The computing theory courses, CSCI 2610 and 2670;
- Data Structures (CSCI 2720), the capstone of the CS Area F and prerequisite for all upper-division CS courses;
- An "Applications design" course, one of CSCI 4050 or 4370;
- Computer Architecture, CSCI 4720;
- A "Systems Design" course, one of CSCI 4570, 4730, or 4760.

Because all students visit these "landmarks," measurements based on these class will be more robust and easier to validate. Because we consider these our most important courses, we have greater confidence that our measurements measure the most important variables.

For each of these ten courses, the faculty will define a short list of key learning objectives that should be met no matter who teaches the course. In Appendix A, we outline how these course objectives map to the undergraduate program objectives from section A above. These are the main measured variables in the model.

C. Measurement instruments: For each offering of the ten landmark CS courses, the extent to which the course met the specific course learning objectives will be measured in three independent ways:

1. A student survey, separate from the course evaluations, will gather the students' opinions on how well the course met each of its learning objectives.
2. The instructor will evaluate how well the course handled each objective.
3. Representatives from the Assessment Committee will examine a stratified sample of student assignments or tests and rate them on how well they handled the objectives. The sample will be chosen to cover all the course-specific objectives and be stratified into A, B, and C students.

These ratings will use a coarse-grained scale with no more than four levels, e.g. "Inadequate", "Expected for a lower-division student", "Expected for a UGA CS graduate," "Superior".

The overall ratings for each program objective will then be computed as a weighted linear combination of scores for appropriate course-specific objectives.

D. Analysis: Using statistical software we will (1) attempt to validate the model by comparing the three independent measurement instruments for consistency, (2) identify problems requiring immediate attention, (3) make longitudinal comparisons to measure our progress, and (4) generate charts and tables to make these results intelligible both to ourselves and the higher Administration.
E. Feedback: The Assessment Committee will compile statistics and prepare most of the text of the required reports, but the department chair should be involved in interpreting the trends and adding the final gloss. The Curriculum Committee will review these documents and propose changes in curriculum or teaching methods as indicated.

Internally, the Assessment Committee will screen the results after each semester to identify acute problems. The individual instructors, the curriculum committee, or the entire Department will deal with these problems, depending on the seriousness of the changes involved.

3. TIMELINE:

This model must be submitted by Dec 16, 2002. In December 2002 and January 2003 the teaching faculty in each "landmark" course will meet and the specific course objectives will be defined. During that period we will also create the linear models for each of the program outcomes.

At the end of Spring 2003 we will conduct a full series of measurements and prepare our first internal assessment report.
APPENDIX A: Relationship of course objectives to program outcomes

Modal standard representing the average level of achievement

From: ACM Curriculum Committee 2001 recommendations, Steelman draft


1. Demonstrate a sound understanding of the main areas of the body of knowledge and the theories of computer science, with an ability to exercise critical judgment across a range of issues.

Courses CSCI 2670, 2720, 4050, 4370, 4570, 4730, and 4760 each cover a specific, key element of the body of knowledge of computer science. Our course-specific measures in these courses will provide broad coverage of how well our students have learned the body of knowledge.

2. Critically analyze and apply a range of concepts, principles, and practices of the subject in the context of loosely specified problems, showing effective judgment in the selection and use of tools and techniques.

This outcome is measured by the students' design choices for their programming projects in upper-division classes. Which technologies they choose for a specific problem is a good indicator of their overall understanding of both the tools and the broad domain of software problem solving.

3. Produce work involving problem identification, analysis, design, and development of a system, along with appropriate documentation. The work must show a range of problem solving and evaluation skills, draw upon supporting evidence, and demonstrate a good understanding of the need for quality.

Students need to learn, practice, and refine these skills in every class that involves computer programming, i.e. all but two of the "landmark" classes. This is an especially critical outcome, because it relates directly to our students suitability for the software development workforce.

4. Demonstrate the ability to work as an individual with minimum guidance and as either a leader or member of a team.
This skill is also developed in our computer programming classes, and is also critical to job success. Our performance on outcomes 3 and 4 is a key indicator of overall program success.

5. Follow appropriate practices within a professional, legal, and ethical framework.

6. Identify mechanisms for continuing professional development and lifelong learning.

Outcomes 5 and 6 are not well supported by our current curriculum. As part of the Assessment program, we will review our curriculum to addressing these issues.

7. Explain a wide range of applications based upon the body of knowledge.

All instructors in our courses will be encouraged to draw the connections between their course content and other applications of computer science. To measure this outcome, we will introduce questions on the student surveys and the instructor feedback.