Agenda

1. [Mike] Votes on items 2, 3 from previous meeting.

2. [Mike, Tom] Discuss TA allocation policy.

3. [Mike] New joint Robotics minor. Discuss, vote.

4. [Jason] Piazza vs. Campuswire, and other options for on-line discussions.

5. [Nelson] More careful deployment of auto-grading technology. Many students don't know how to run tests themselves; rather, they over-rely on the submit server to do testing for them.

6. [Bill G and Ramani (their TAs)] Gradescope for auto-grading.

Details

1. [Mike] Recall:

- (2) Change requirements -- decouple 216 and 250 so they are not co-reqs. Limit elective credits to only allow one 499A. Limit elective credits to at most 3 credits of 388/389X courses.

- (3) Re-consider approach to granting permission for 400-level courses. Right now, the course permission process for all 400 level courses is as follows:
    A. Undergrad CS and CE students, grad CS students get first pick at courses at the time of registration - grad students get permission from Tom.
    B. All non-majors fill out a form to request courses and are granted permission on or around the first day of the semester. Undergrads must meet prereqs, and grad students take the course at their own risk. The undergrad office administers the stamps.

2. [Mike, Tom] Working document. Here is the TA numbers cheat sheet as well. Slide presentation included.

3. [Mike] See linked document

4. [Jason] CampusWire is better than Piazza. We should move to it. Can I get an 'amen' ?

5. [Nelson] More careful deployment of auto-grading technology. The system we use

in 131/132/216 of public, release tests is causing the problem that students don't know how to run tests themselves; rather, they over-rely on the submit server to do testing for them. Students use public tests as a slot machine in Vegas; they run them, change code, run them again, until the pass the tests. This semester I am trying to address this matter by asking students to look at the tests (see http://www.cs.umd.edu/~nelson/classes/resources/javadebugging/).

Possible solutions to this problem:

a) No public tests; minimal amount of driver(s) with input and expected output (not even JUnit public tests).
b) Release tests should be checking for syntax of expected classes and methods and not for all or large portion of the functionality.
c) Most of the grade should come from secret tests (at least 60 percent).
d) Each course can start with public tests, but less information is provided as the course moves from one project to the next.

6. [no additional docs]