Curriculum Committee Minutes

May 11, 2007

Attendees: Tom Doeppner, Chad Jenkins, Claire Kenyon, Franco Preparata, Steve Reiss, John Savage (Chair)

Guest: John Jannotti

1. The CS32/36 Requirement for Systems Course Pairs

The committee discussed a request by students that we change the current requirement for both CS32 and CS36 when students take a course pair for AB or ScB concentration that includes a systems course. Students have expressed themselves in a petition with 48 signatories.

Four options were considered and discussed. They are 1) no change in requirements; 2) keep the two courses without change but drop the requirement that both be taken for systems course pairs; 3) revise CS32 so that it contains material from both CS32 and CS36, in particular, so that instruction in C and C++ is given; 4) bring back CS34, a half-credit course providing an introduction to C and C++; or 5) revise CS32 as indicated above but retain a revised CS36 as a service course. Steve presented a revised CS 32, which is attached, that represents the collective opinion of the systems faculty.

Polling indicated the following interest in these options.

<table>
<thead>
<tr>
<th>Option</th>
<th>Yes</th>
<th>No</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) No change</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>2) Drop requirement for both CS32 and CS36</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>3) Revise CS32, drop CS36</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>4) Bring back CS34</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>5) Revise CS32, keep CS36</td>
<td>6</td>
<td>0</td>
</tr>
</tbody>
</table>
As a result of subsequent discussion we decided that the preferred option is the fifth with the understanding that CS36 would probably need revision both because of potential overlap with a revised CS32 and because it would now serve a different purpose. It was agreed that Tom Doeppner, Pascal van Hentenryck and Shriram Krishnamurthi, all of whom have expressed an interest in this course, be asked to meet and make recommendations on its disposition.

Because the changes under consideration would affect most of our concentrations, it was also decided that we should refer this matter, after a revised CS36 is prepared, to the full faculty for action. This is a matter on which action should be taken early so that students can plan for the change.

2. Content of a Revised CS32

A discussion in depth of the proposed revision of CS32 was held. Steve said that the changes removed six lectures on design and specification that would be moved to evening sessions and labs and instruction on C and C++ added. He reports that the workload would be about the same. The objective it keep the number of hours needed in the course to less than 15 per week. John Jannotti said that he would be interested in teaching a revised CS32 and was very satisfied with the proposed description.

We briefly discussed, without reaching a conclusion, how the transition from the old to new requirements might be made. We discussed the possibility of allowing students who had taken CS36 but not CS32 to substitute it for CS32. We also considered the possibility that students could substitute CS190 for CS32. This subject needs further discussion.

CS32: Introduction to Software Engineering

The purpose of this course is to provide students with the background they need to design and implement moderate (10-50k loc) software systems. The emphasis of the course is on learning the design and coding techniques that are needed in the implementation of such systems. Design techniques include understanding design from the class level through to software architectures, understanding how to express designs, understanding how to design user interfaces as part of such systems, understanding how to use threads and sockets, understanding how to use data and databases, and understanding how to integrate applications with the web. Coding techniques include proper use of tools, an emphasis on testing, memory management, proper use of
programming languages, and coding and programming styles. The course also serves as a basic introduction to software engineering, with students doing a team project from requirements through implementation. Here emphasis is on learning how to design for and work as a team, doing presentations, and that the primary goal when writing code is to enable it to be maintained. Prior to the final project students have the experience of modifying their earlier code and of doing a small team effort.

WEEK 1:

Introduction to the course
Hand out first assignment
Software engineering overview (phases, purpose)

WEEK 2:

JUnit and testing first (lab)
Interface design.
Class design.
Package design.
UML as a tool and way of expressing designs
Project team selection.
Hand out second assignment/ design due for first assignment
(Second assignment is modification of the first)
Project Teams

WEEK 3:

Eclipse, debugging (lab)
XML
Requirements
Introduction to threads
First assignment due / design for second assignment due

WEEK 4:

Threads in Java (lab)
Designing with threads; synchronization in Java
Sockets in Java
Second assignment due
Project requirements due
Specifications
Third assignment out
(Third assignment involves server handling multiple clients; also is a 2-person project; also requires measurement of throughput)

WEEK 5:
Configuration management (lab)
Using sockets with threads
HTML/URL/HTTP and how these work
Client-server designs
Third assignment design due
Project specifications due
Project specification presentations (spec day)

WEEK 6:
Socket programming (lab)
Core+extension designs
Event-based programming; publish-subscribe
User interface toolkits
User interface design
Third assignment due

WEEK 7:
Swing programming (lab)
DBMS (SQL/JDBC)
Web application design
Scripting languages for the web
Project design due
Design presentations given in special session
(design day; use labs?)

WEEK 8:
Database programming (lab)
Java behind the scenes
Memory management
JNI
Initial project prototype due

WEEK 9:
Project prototype demonstrations (lab)
C and Unix programming
Memory allocation
Fourth assignment out (C programming; simple)

WEEK 10:
  SPRING BREAK

WEEK 11:
  Compiling C, Memory allocation (lab)
  C++ programming
  How to use C++ safely
  Fourth assignment due
  Fifth assignment out (C++ programming/JNI)

WEEK 12:
  Compiling C++, C++ tools (lab)
  Advanced C++ programming
  STL and data structures
  Template programming

WEEK 13:
  STL programming (lab)
  Threads and sockets in C and C++
  (posix; creating appropriate classes)
  Synchronization primitives in C/C++
  Fifth assignment due
  Project C level

WEEK 14:
  Project demos (lab)
  File I/O, mmap
  Select and poll
  Linking, loading, dynamic libraries
  Project B level

WEEK 15:
  Human Factors
  RPC, remote objects, SOAP
  Final project predemos (lab)
WEEK 16:

Project demos (demo day)

Issues:

Students won't become adept C/C++ programmers from this course. They will know the language and understand some of how to program it.

Project is more spread out. It might be difficult to do a larger project within the time constraints. Also some aspects of the project (e.g. networking, sockets) might not be taught or understood at the time the project needs to be designed. Note that they still have 5 weeks dedicated exclusively to the project (i.e. with no other programming assignments).

What do we do with current students who have take one or the other of the courses?

This looks like a more intense course. CS32 had gotten down to 10-15 hours a week; this looks more like 15-20.

The course should have a scheduled lab session. This occurs throughout the course (not just in the first half). The idea is to teach some of the concepts hands-on rather than in lecture. There probably should also be sections going over specific material needed for the homework or providing review as needed.

For next year we could have an alternative assignment in place of C and C++ that would be akin to DnaSeq or CdQuery for those who have taken 36 already.