UW/CSE Systems Programming — CSE 333

Course Description

Draft of September 14, 2009

Structural place in the curriculum

- 4 credits (3 weekly lectures, 1 weekly section, no lab)
- Pre-requisites: Hardware/Software Interface
- Subsequent courses: The following courses would have this course as a pre-requisite: Computer Graphics, Computer Vision, Software for Embedded Systems. The following courses would have this course as recommended or strongly recommended: Operating Systems, Networks. It would also be useful to students in Databases, Compilers, and similar areas.
- Taken by: Required for CE hardware track and students in other courses that list it as a prerequsite; elective for all other CS and CE students.
- Catalog description: To be determined

Course Overview / Goals

The focus of this course is techniques for designing, developing, and debugging programs in C and C++, including programs that have multiple source files, communicate with the operating system, and potentially do asynchronous or parallel computation and I/O. The course would include tools like debuggers and Makefiles that are needed for effective programming in this environment, but that would not be the core intellectual component of the course. The focus would be programs that are "closer to the machine" than those in the Software Design and Implementation course, which would be complementary to the material covered here.

A goal of this course is to have a single 300-level course that would provide introductory C/C++ competency so that relevant 400-level courses could focus on the material important to their particular area and not have to cover the basics taught here.

Description of Possible Homework, Etc.

The bulk of the work in this course would consist of substantial programming projects implemented in C and C++. The major projects would be supplemented by written homework exercises as needed. This course would not have a lab component.

A challenge for this course is to have assignments and projects that are well-motivated but do not overlap detailed content in later courses.

Possible Textbook

To be supplied.

(Most likely instructor preference. There do not seem to be any obvious textbooks for the course. Instructors will probably want to use some combination of written and online tutorial and reference materials.)

Approximate Topic List

This list gives the approximate amount of time needed for various topics. The items here will not necessarily be presented in this sequence. For example, it is helpful to intersperse material about development tools between the language and programming topics so they are introduced when they are most relevant for assignments.

[This needs to be checked for overlap with topics covered in the Hardware/Software Interface and Software Design & Implementation courses and adjusted accordingly.]

- C programming (2 weeks)
  - Review of basic C programming and memory model from HW/SW interface course
  - Pointers, lvalues & rvalues, structs, casts
  - Arrays and strings, 2-D arrays with and without pointers
  - Dynamic storage allocation (malloc/free)
  - C preprocessor, multifile programs
  - Core C libraries (I/O, strings, etc.)
  - Idioms for error handling without exceptions
- Essential tools for C/C++: compilers, debuggers, make, version control, etc. (1 week)
- Memory management and system interface (1 week)
  - Idioms for manual memory management; avoiding dangling pointers and memory leaks
  - Memory management implementation
  - Linking and libraries — how a program is assembled
  - Relation between libraries and underlying OS services
- C++ programming (3 weeks)
  - Basic C++: "a better C", C with classes
  - Class definitions, constructors, copy constructors, destructors, const, other details
  - Dynamic memory allocation (new/delete), classes with dynamic data
  - Classes and inheritance in C++; overloading, overriding
  - Using C++ templates and STL
- Best practices (1 week)
  - Class design and patterns in C++
  - Systematic program development and debugging
  - Profiling and optimization
- Threads and concurrency in C/C++ (1 week)
  - Concurrent programming beyond HW/SW interface course
  - Programming in a multicore world
  - Asychronous I/O, networking, and user interfaces
- Security issues in C/C++ ($<$ 1 week specifically, but discussed throughout where appropriate, e.g., safe vs unsafe library functions)

Additional Topics
These are topics that are "close" to the course, but that do not likely fit in 10 weeks. If our time estimates are wrong or some topics above are deemed unimportant, they would be fine additions.

- Networking
- GUIs and user interface issues

- Automatic garbage collection vs manual memory management
- Others? ...

Background / Correspondence to Old Curriculum
The C/C++ programming material is an expanded version of that currently in CSE 303. Unlike that course, there is minimal coverage of the Unix/Linux toolchain, shell scripting, utilities, or social issues. Instead there is more substantial treatment of the major programming issues, and the hope is that after taking this course students would be significantly more competent developing C/C++ software than they are after the more limited treatment currently in CSE 303.