8/16/09

Appendix C Programming and Operating System Projects

Many instructors believe that implementation or research projects are crucial to the clear understanding of operating system concepts. Without projects, it may be difficult for students to grasp some of the basic OS abstractions and interactions among components; a good example of a concept that many students find difficult to master is that of semaphores. Projects reinforce the concepts introduced in this book, give the student a greater appreciation of how the different pieces of an OS fit together, and can motivate students and give them confidence that they are capable of not only understanding but implementing the details of an OS.

In this text, I have tried to present the concepts of OS internals as clearly as possible and have provided numerous homework problems to reinforce those concepts. Many instructors will wish to supplement this material with projects. This appendix provides some guidance in that regard and describes support material available at the instructor's Web site. The support material covers eight types of projects and other student exercises:

- Animations
- Simulation projects
- Programming projects
- Research projects
- Reading/report assignments
- Writing assignments
- Discussion topics
- Commentary on BACI and NACHOS

C.1 ANIMATIONS

Animation provides a powerful tool for understanding the complex mechanisms of a modern OS. Today's students want to be able to visualize the various complex OS mechanisms on their own computer screen. The sixth edition incorporates 16 separate animations covering such areas as scheduling, concurrency control, cache coherency, and process life cycle. Table C.1 lists the animations by chapter. At appropriate places in the textbook, the animations are indicated by an icon, so that the student can invoke the animation at the proper point in studying the book. The animations will be made available to professors at the **Instructor's Resource Center (IRC)** for this book in such a way as to enable online access by students.

C.2 SIMULATIONS

The IRC also provides support for assigning projects based on a set of simulations developed at the University of Texas, San Antonio. Table C.2 lists the simulations by chapter. The simulators are all written in Java and can be run either locally as a Java application or online through a browser.

The IRC includes the following:

1. A brief overview of the simulations available.
2. How to port them to the local environment.
3. Specific assignments to give to students, telling them specifically what they are to do and what results are expected. For each simulation, this section provides one or two original assignments that the instructor can assign to students.

These simulation assignments were developed by Adam Critchley (University of Texas at San Antonio).

C.3 PROGRAMMING PROJECTS

Three sets of programming projects are provided.

Textbook-Defined Projects

Two major programming projects, one to build a shell, or command line interpreter, and one to build a process dispatcher are described in the textbook, after Chapter 3 and after Chapter 9. The IRC provides further information and step-by-step exercises for developing the programs.

These projects were developed by Ian G. Graham of Griffith University, Australia.

Additional Major Programming Projects

A set of programming assignments, called machine problems (MPs), are available that are based on the Posix Programming Interface. The first of these assignments is a crash course in C, to enable the student to develop sufficient proficiency in C to be able to do the remaining assignments. The set consists of nine machine problems with different difficulty degrees. It may be advisable to assign each project to a team of two students.

Each MP includes not only a statement of the problem but a number of C files that are used in each assignment, step-by-step instructions, and a set of questions for each assignment that the student must answer that indicate a full understanding of each project. The scope of the assignments includes:

1. Create a program to run in a shell environment using basic I/O and string manipulation functions.
2. Explore and extend a simple Unix shell interpreter
3. Modify faulty code that utilizes threads.
4. Implement a multithreaded application using thread synchronization primitives.
5. Write a user-mode thread scheduler
6. Simulate a time-sharing system by using signals and timers
7. A six-week project aimed at creating a simple yet functional networked file system. Covers I/O and file system concepts, memory management, and networking primitives.

The IRC provides specific instructions for setting up the appropriate support files on the instructor's Web site of local server.

These project assignments were developed at the University of Illinois at Urbana-Champaign, Department of Computer Science and adapted by Matt Sparks (University of Illinois at Urbana-Champagne) for use with this textbook.

Small Programming Projects

The instructor can also assign a number of small programming projects described in the IRC. The projects can be programmed by the students on any available computer and in any appropriate language: They are platform and language independent.

These small projects have certain advantages over the larger projects. Larger projects give students more a sense of achievement, but students with less ability or fewer organizational skills can be left behind. Larger projects usually elicit more overall effort from the best students. Smaller projects can have a higher concepts-to-code ratio, and because more of them can be assigned, the opportunity exists to address a variety of different areas. Accordingly, the instructor's IRC contains a series of small projects, each intended to be completed in a week or so, which can be very satisfying to both student and teacher. These projects were developed by Stephen Taylor at Worcester Polytechnic Institute, who has used and refined the projects in the course of teaching operating systems a dozen times.

C.4 RESEARCH PROJECTS

An effective way of reinforcing basic concepts from the course and for teaching students research skills is to assign a research project. Such a project could involve a literature search as well as a Web search of vendor products, research lab activities, and standardization efforts. Projects could be assigned to teams or, for smaller projects, to individuals. In any case, it is best to require some sort of project proposal early in the term, giving the instructor time to evaluate the proposal for appropriate topic and appropriate level of effort. Student handouts for research projects should include

- A format for the proposal
- A format for the final report
- A schedule with intermediate and final deadlines
- A list of possible project topics

The students can select one of the listed topics or devise their own comparable project. The instructor's Web site includes a suggested format for the proposal and final report as well as a list of possible research topics developed by Professor Tan N. Nguyen of George Mason University.

C.5 READING/REPORT ASSIGNMENTS

Another excellent way to reinforce concepts from the course and to give students research experience is to assign papers from the literature to be read and analyzed. The IRC site includes a suggested list of papers to be assigned, organized by chapter. The IRC provides a copy of each of the papers. The IRC also includes a suggested assignment wording.

C.6 WRITING ASSIGNMENTS

Writing assignments can have a powerful multiplier effect in the learning process in a technical discipline such as OS internals. Adherents of the Writing Across the Curriculum (WAC) movement (http://wac.colostate.edu/) report substantial benefits of writing assignments in facilitating learning. Writing assignments lead to more detailed and complete thinking about a particular topic. In addition, writing assignments help to overcome the tendency of students to pursue a subject with a minimum of personal engagement, just learning facts and problem-solving techniques without obtaining a deep understanding of the subject matter.

The IRC contains a number of suggested writing assignments, organized by chapter. Instructors may ultimately find that this is an important part of their approach to teaching the material. I would greatly appreciate any feedback on this area and any suggestions for additional writing assignments.

C.7 DISCUSSION TOPICS

One way to provide a collaborative experience is discussion topics, a number of which are included in the instructor's supplement. Each topic relates to material in the book. The instructor can set it up so that students can discuss a topic either in a class setting, an online chat room, or a message board. Again, I would greatly appreciate any feedback on this area and any suggestions for additional discussion topics.

C.8 BACI AND NACHOS

In addition to all of the support provided at the IRC, there are two publicly available packages that instructors may wish to use:

- **Ben-Ari Concurrent Interpreter (BACI):** BACI simulates concurrent process execution and supports binary and counting semaphores and monitors. BACI is accompanied by a number of project assignments to be used to reinforce concurrency concepts.
- **Nachos:** Nachos is a simulated OS environment suitable for generating implementation projects appropriate for an introductory course in OS design. Nachos can be used to reinforce OS concepts and is accompanied by a number of project assignments.

This appendix provides a brief discussion of these topics. Appendix H provides a more detailed introduction to BACI, with information about how to obtain the system and the assignments. Nachos is well documented at its Web site and is described briefly in this section.

Nachos Overview

Nachos is an instructional operating system that runs as a UNIX process, to provide students with a reproducible debugging environment, and that simulates an operating system and its underlying hardware [CHRI93]. The goal of Nachos is to provide a project environment that is realistic enough to show how real operating systems work yet simple enough that students can understand and modify it in significant ways.

A free distribution package is available via the Web that includes

- An overview paper.
- Simple baseline code for a working operating system.
- A simulator for a generic personal computer/workstation.
- Sample assignments: The assignments illustrate and explore all areas of modern operating systems, including threads and concurrency, multiprogramming, system calls, virtual memory, software-loaded TLBs, file systems, network protocols, remote procedure calls, and distributed systems.
- A C++ primer (Nachos is written in an easy-to-learn subset of C++, and the primer helps teach C programmers this subset).

Nachos has been used at hundreds of universities around the world and has been ported to numerous systems, including Linux, FreeBSD, NetBSD, DEC MIPS, DEC Alpha, Sun Solaris, SGI IRIX, HP-UX, IBM AIX, MS-DOS, and Apple Macintosh. Future plans include a port to Stanford's SimOS, a complete machine simulation of an SGI workstation.

Nachos is freely available from its Web site (there is a link to the Web site from WilliamStallings.com/OS/OS6e.html); a solution set is available to instructors by e-mail from nachos@cs.berkeley.edu. In addition, there is a mailing list for instructors and a newsgroup (alt.os.nachos).

Choosing between Nachos and BACI

If the instructor is willing to take the time to port one of these simulators to the local environment available to the students, then the choice between will depend on the instructor's objectives and personal opinion. If the focus of the projects is to be on concurrency, then BACI is the clear choice. BACI provides an excellent environment for studying the intricacies and subtleties of semaphores, monitors, and concurrent programming.

If, instead, the instructor wishes to have students explore a variety of OS mechanisms, including concurrent programming, address spaces and scheduling, virtual memory, file systems, networking, and so on, then Nachos may be used.