SPRING 1993
COMPUTER SCIENCES DEPARTMENT
UNIVERSITY OF WISCONSIN—MADISON
PH. D. QUALIFYING EXAMINATION

Operating Systems
Breadth Examination

Wednesday, February 10, 1993
3:00 – 5:00 PM
113 Psychology

GENERAL INSTRUCTIONS:

1. Answer each question in a separate book.
2. Indicate on the cover of *each* book the area of the exam, your code number, and the question answered in that book. On *one* of your books list the numbers of *all* the questions answered. *Do not write your name on any answer book.*
3. Return all answer books in the folder provided. Additional answer books are available if needed.

SPECIFIC INSTRUCTIONS:

Answer any *four* of the following *six* questions. Before beginning to answer a question, be sure to read it carefully. If you are confused about what the question means, state any assumptions you have made in formulating your answer. Good Luck!

POLICY ON MISPRINTS AND AMBIGUITIES:

The Exam Committee tires to proofread the exam as carefully as possible. Nevertheless, the exam sometimes contains misprints and ambiguities. If you are convinced a problem has been stated incorrectly, mention this to the proctor. If necessary, the proctor can contact a representative of the area to resolve problems during the *first hour* of the exam. In any case, you should indicate your interpretation of the problem in your written answer. Your interpretation should be such that the problem is nontrivial.

Operating Systems

Instructions:
For the **BREADTH** exam: Answer any 4 of questions 1-6.
For the **DEPTH** exam: Answer any 4 of questions 1-6 *and* 3 of questions 7-10. You are expected to spend about two hours on each half of the exam.

BREADTH QUESTIONS

Breadth 1. Working Set:

The *Working Set* model of page replacement has a single parameter, τ.

(1a) What is this parameter? How does it differ from parameters in other place replacement algorithms?

(1b) What factor(s) affect your choice of value for τ, to what extent, and why?

(1c) What value would be an initial good guess for τ?

Breadth 2. Name Services:

Databases usually require that all sites are updated consistently; that is, queries are atomics and appear as if they were globally serialized. Name Services usually have a much weaker notion of consistency.

(2a) What is the consistency guarantee in Grapevine?

(2b) What is the motivation for a naming services to provide such a weak notion of consistency?

Breadth 3. Monitors:

The operating system usually associates a queue with each monitor *condition* variable. When a process performs a `wait(c)`, the processes is blocked and placed on the queue associated with `c`. A process is unblocked using a call to `signal(c)`. Typically, these queues unblock processes in FIFO order.

(3a) Is it possible for a programmer *using* monitors to write code that can detect whether a condition variable queue is using FIFO order? Explain.

(3b) Does your answer depend on whether the monitor is using Hoare or Mesa semantics? Explain.

Breadth 4. Copy-on-Write:

Copy-on-write is a way for processes to share pages (or segments) that are logically distinct. When one process sends some data to a second process with copy semantics, nothing needs to be copied immediately, or perhaps ever.

(4a) Explain how standard memory-management hardware can be used to implement copy-on-write cheaply.

(4b) Describe how this method can be used to improve system performance for:

- (i) message passing,
- (ii) forking a new process, and
- (iii) file I/O operations.

Breadth 5. Messages:

Consider message-based interprocess communications. In a ‘‘send’’ operation, there are several ways to identify (name) the recipient of the message.

Describe three (3) ways, listing the advantages and disadvantages of each.

Breadth 6. Disk Organization:

In the “classic” Unix file system, the contents of a file are stored in a set of small blocks allocated more or less randomly from a free list.

(6a) The Berkeley “fast file system” changed the representation of files to improve the performance of a single process sequentially reading or writing a single large file. Briefly outline those changes that are specifically related to improving performance of this kind of access.

(6b) Suppose you have to design a new file system for a uniprogramming environment to support a single dedicated application. >From time to time this application creates or destroys files, but most of the time, it is sequentially reading one file at a time (with the choice of which file to read a uniform random variable). The application only runs 9 to 5 on weekdays. Design an appropriate file organization for this system.