From: mendel@Lagunita.Stanford.EDU
To: shankle@ee.stanford.edu (Diane Shankle)
Subject: Re: Quals Questions
Date: Mon, 20 Mar 95 02:45:05 PST

EE Quals - Winter 1995 - Mendel Rosenblum

1)
One of the things that operating systems do is manage the hardware resources of a computer system. Examples of resources include storage space (memory, disks), processing elements (CPU, I/O devices), higher level abstractions (open files, network connections, etc).

There are two related techniques used to control resources: scheduling and allocation.

(a) What's the difference between these techniques?

It is important for the OS not to "lose" track when allocating storage. Two different techniques are used. One is to have the OS explicitly track of storage allocation and deallocation. The other is for the OS not to explicit track but it might need to search around and find requests when needed.

(b) Describe the tradeoffs between these techniques. Give examples when one is better. When would you use each.

ANSWER #1
(a)
We usually used scheduling when talking about preemptable resources and allocation when talking about non-preemptable resources. Scheduling implies "How long" while allocation is "Too whom".

(b)
This is basically asking you to compare explicit resource management such as reference counting with garbage collection. If you have plenty of resources so that you rarely run out, garbage collection can result in a lower overhead than explicitly tracking storage. Explicit is good when tight controls are needed.

2)
The OS defines the interface to resources and services. This can be thought of as an user interface for application programmers. It allows user access and control of resources. Describe issues (tradeoffs) in what should go into interfaces. What do we look for in such a interface. (goals?)

(Example: Subsystem FS)

(b) Are the differences between interfaces for multiprogramming system and personal computers? What? (Optimizations?)

ANSWER #2
When designing such an interface you need to tradeoff ease of use with providing information for optimizations. More information can make the OSes job easier but the programmers harder.

(b)
Typically, more information is needed in a multiprogramming system to handle balancing the system between the users.