Computer Science 340

Operating Systems

FIRST TEST, 1994

READ THIS FIRST !!!!!!!!!!!!!

Answer all questions -
- BRIEFLY. I don't think that you should need more than one side of paper for question 1 and two for question 2. The two sheets of paper attached to the question sheet should be sufficient.

Make sure your name is on every piece of paper which you hand in.

Please write on the top of the front page the coordinates whch define your position in the test room. (You will be told what they are near the beginning of the test.)

The total mark for question 1 is half the total mark for question 2.

The marks for each question are equally divided between the parts of the question.

There are TWO questions; the test lasts for 45 minutes. As I imagine you can work out for yourself, that can reasonably be interpreted as about 15 minutes for question 1 and 30 minutes for question 2.

I expect answers in terms of the material already covered in the 340 course, together with such general computing knowledge as can reasonably be expected of a stage 3 Computer Science student, which includes general knowledge of the Macintosh and Unix systems; essays on topics not yet treated in the course are not required and will not receive any marks.

I try to give all the information you need to answer each question, but don't always succeed. If you consider that you have insufficient information to answer a question, don't ask a question in the test : explain your difficulty in your written answer, say what further information you would need to resolve it, and make clear any assumptions that you have made.

This is an open-book test.

QUESTION 1.

It is claimed that hardware assistance is needed to implement an effective protection system in any computer shared by two or more people. Two hardware features are commonly provided for this purpose :

- memory address checking, typically provided by base and limit registers or some equivalent, and

- a processor supervisor mode in which certain privileged machine instructions are available, usually entered by a supervisor call operator which both executes a branch to a fixed machine address and switches the processor from normal to supervisor mode.

Assume in your answer that the processor in the system you are discussing has no other unusual characteristics.

(a) To what extent are these features necessary in protecting a process's primary memory from interference by another process ? (Bear in mind that the system must continue to be sharable; schemes in which one programme monopolises the computer for ever are not acceptable.)

(b) To what extent can the same features be used in protecting disc files against access by programmes not belonging to their owners ?

(c) It has also been claimed that encryption is as effective as hardware as a basis for protection systems. It is argued that it doesn't matter whether or not an intruder can read other people's material provided that the material is impossible to decode. Is that true ? (- and give reasons !)

(CONTINUED ON THE NEXT PAGE.)

QUESTION 2.

It is proposed to construct a TUI (which stands for textual user interface - and note the local colour). The TUI consists of hardware (the TUI terminal - a keyboard, mouse (with no button), and screen) and software (the user interface management system, or UIMS). The UIMS runs on its own processor, and communicates with processes running in other computers by exchanging messages in the form of character strings carried on a single serial communications line. This question is about the function of the UIMS.

The TUI displays text in windows on a screen. The UIMS should handle all details of dealing with the keyboard and mouse and managing the screen display, but should otherwise be *as simple as possible*, consistent with the requirements listed in the specification at the end of the question. READ THE SPECIFICATION : it is intended to simplify the question, not to complicate it. Most of it follows the Macintosh conventions where it makes sense, so will be familiar to you.

Question :

NOTE : Think about the whole question before committing yourself to an answer. The parts are to some extent dependent on each other, and I shall expect a consistent answer.

(a) What data must the UIMS maintain to manage the screen adequately ? Describe any data structures used in sufficient detail to answer part (b) satisfactorily.

(b) What must the UIMS do with its data structures in order to react correctly to
(i) a character entered at the keyboard;
(ii) a movement of the mouse;
(iii) a character arriving from a process ?

(Just say what the UIMS does - DO NOT try to write code. Pseudocode is acceptable.)

(c) Design a set of messages which should be accepted or transmitted by the UIMS for communication with processes during normal operation.

Assume :

Interface errors need not be considered. (No errors occur within the interface; all messages arriving from the computer are correct.)

Some other device outside the TUI is responsible for collecting messages from processes to the TUI and distributing messages sent by the TUI to the processes. This device is not your concern - just make sure that it has all the information which it needs to do its job.

SPECIFICATION

TUI COMPONENTS.

The TUI mouse has no button. It is used (as in the Macintosh system) to control the position of a pointer on the screen. Signals from the mouse are sent to the UIMS; there is no magical link between the mouse and the pointer on the screen.

Text entered at the TUI's keyboard is sent to the UIMS, which must display it in the currently active window, and also send it to the process which constructed the window, identified with the window number.

The TUI screen is a rectangular character display. All character positions are the same size and are addressed by their horizontal and vertical coordinates. The displayed characters are generated by hardware given the coordinates and the ordinary ASCII character code.

(CONTINUED ON THE NEXT PAGE.)

Communication between UIMS and processes is carried out by transmitting and receiving messages. The messages may be of any length, but must contain sufficient structure to be decipherable by their recipients.

SCREEN.

The screen is organised into windows. The full screen is considered to be a window owned by the UIMS, and processes may construct new windows and destroy windows which they have constructed. A process may have any number of windows. For this purpose, the UIMS is considered to be a process.

The information in each window is in the form of text strings, every text string is in a window, and every window contains exactly one text string. The text string of a new window is a null string, containing no characters.

The string is displayed in the window as on a page of (English) text, split into lines as required to fit into the window. If the string fills the window, the text is "scrolled" up the window line by line so that the beginning of the string is lost, and the end of the string is on the last line of the window.

A pointer is displayed on the screen. Its position is controlled by the UIMS. The pointer is never hidden.

PROCESSES.

Every process is identified by a unique process number, allocated outside the TUI; the UIMS's process number is 1.

WINDOWS

Windows may be of any size which will fit on the screen, and may overlap. The position and size of a window may not be changed. To construct a window, the UIMS must know its top, bottom, left, and right coordinates.

Windows are ordered, from back to front. When windows overlap, windows further forward hide windows further back. The full screen is always the back window. If a new window includes the current pointer position, it is made the front window; otherwise it becomes the window following the front window. The UIMS may change the order of the windows. Notice that the order of the windows is quite distinct from the window numbers.

The window in which the pointer lies is defined to be the currently active window, and it remains the currently active window until the pointer is moved over its boundary. Unless the currently active window is the full screen, it is made the front window, and is therefore displayed in full.

Text for any window may be presented to the UIMS by processes external to the TUI. The text is always copied immediately to the requested window (though if the window is obscured the text may not appear immediately on the screen), and is always appended to the end of the window's text string.

The UIMS may only make new windows or dispose of old ones on explicit instruction from a process.

Alan Creak,
April, 1994.

(CONTINUED ON THE NEXT PAGE.)