THE AUSTRALIAN NATIONAL UNIVERSITY

Second Semester 2003

COMP2310
(Concurrent and Distributed Systems)

Writing Period: 3 hours duration

Study Period: 15 minutes duration

Permitted Materials: None

All your answers must be written in the boxes provided in this booklet. You will be provided with scrap paper for working, but only those answers written in this booklet will be marked. Do not remove this booklet from the examination room. There is additional space at the end of the booklet in case the boxes provided are insufficient. Label any answers you write at the end of the booklet with the number of the question they refer to.

Greater marks will be awarded for answers that are simple, short and concrete than for answers of a sketchy and rambling nature. Marks will be lost for giving information that is irrelevant to a question.

Name (family name first):

Student Number:

Official use only:

Q1 (25)	Q2 (25)	Q3 (25)	Q4 (25)	Q5 (25)	Q6 (25)	Total (150)

QUESTION 1 [25 marks]

(a) What standard components of computer architecture make it possible to implement multiple processes (as in Unix, for example) on a single CPU? Explain (a) how the operating system uses these features for this purpose, and (b) what data structures are involved.

QUESTION 1(a)	**[5 marks]**

(b) Suppose `fd` is an integer variable whose current value is that of an open file descriptor. Show how, in C/Unix, to start a child process which redirects standard output to `fd` and then invokes the `ls` program (i.e. so that the output of `ls` goes into `fd`).

QUESTION 1(b)	**[5 marks]**

(c) Explain why the central property of pure functional languages makes them of particular interest in the context of multi-processor machines.

QUESTION 1(c)	**[5 marks]**

(d) Characterize and give examples of computing situations that might be described as instances of the *readers and writers problem*. Include in your answer an explanation of *read transactions* and *write transactions*.

QUESTION 1(d)	**[5 marks]**

(e) Explain the design philosophy behind *micro-kernels*. What are their advantages and disadvantages compared with monolithic kernels?

QUESTION 1(e)	**[5 marks]**

QUESTION 2 [25 marks]

(a) Define the term *critical section*, placing it in the context of the *mutual exclusion problem*. Explain, with the aid of an example, what it means for a program to have more than one *class* of critical section. Show how to use the test-and-set instruction to implement your example.

QUESTION 2(a)	**[10 marks]**

(b) Define the semaphore data type, including a description of the behaviour of the `wait` and `signal` operations.

QUESTION 2(b)	**[5 marks]**

(c) Show how to use semaphores to implement a rendezvous between two processes.

QUESTION 2(c)	**[5 marks]**

(d) Compare and contrast the behaviour of Unix (heavyweight) processes and Java (lightweight) threads.

QUESTION 2(d)	**[5 marks]**

QUESTION 3 [25 marks]

(a) What are the specific advantages and disadvantages of monitors over conditional critical regions?

QUESTION 3(a)	**[5 marks]**

(b) Assume you have system available which offers mutual exclusion via semaphores. Can you construct a complete monitor based on these primitives? Explain what you have to consider.

QUESTION 3(b)	**[5 marks]**

(c) Protected objects (as specified in Ada95) do not offer condition variables. What do they offer instead? How does this differ from condition variables?

QUESTION 3(c)	**[5 marks]**

(d) What are the potential problems if a monitor allows for suspension on condition variables in the middle of procedures/methods inside a monitor? How can you deal with them?

QUESTION 3(d)	**[5 marks]**

(e) Why is the use of object-oriented programming potentially risky in conjunction with synchronization primitives?

QUESTION 3(e)	**[5 marks]**

QUESTION 4 [25 marks]

(a) Which constructs in CSP, occam 2.1, and Ada95 are intrinsically non-deterministic? Explain why they are chosen to behave that way, or why they can not be implemented otherwise anyway. Some constructs can be intentionally non-deterministic, while others are non-deterministic by nature, so classify them all and motivate them individually.

QUESTION 4(a)	**[5 marks]**

(b) Explain the notion of a *process* in occam 2.1.

QUESTION 4(b)	**[2 marks]**

(c) The Ada95 language specification is many times larger then the occam 2.1 specification (580 pages of small print specifications for the core Ada95 language; motivation and rationale are separate documents).

What are the principal additional features of Ada95? (Name three or more principal differences.) Classify the items in your answer as 'nice to have (but basically redundant)' or 'essential for ... '. Name specific application areas where occam 2.1 could not be used (or only in a clumsy way) and Ada95 offers easy solutions.

QUESTION 4(c) **[8 marks]**

(d) Syntactically similar constructs can have very different semantics. Compare and contrast the semantics of the `switch` statement in C, the `select` statement in Ada95, and the `CASE` statement in occam 2.1.

QUESTION 4(d)	**[5 marks]**

(e) Can you simulate asynchronous message passing in Ada95? What about occam 2.1? If you think it is possible, explain how you would implement this. Sketch your ideas; there's no need to provide a full implementation.

QUESTION 4(e)	**[5 marks]**

QUESTION 5 [25 marks]

(a) What are the conditions under which a deadlock may occur?

QUESTION 5(a)	**[3 marks]**

(b) What is deadlock avoidance? List possibilities for implementing this.

QUESTION 5(b)	**[2 marks]**

(c) Why is it that many deadlock detection algorithms can only detect the possibility of a deadlock, but not necessarily the existence of a deadlock?

QUESTION 5(c)	**[5 marks]**

(d) What is a safe system (with respect to deadlocks) and how can you achieve such a system?

QUESTION 5(d)	**[5 marks]**

(e) Under which circumstances is deadlock recovery easy? Which problem is not solved by deadlock recovery?

QUESTION 5(e)	**[2 marks]**

(f) Describe the Banker's algorithm. What problem does it solve?

QUESTION 5(f)	**[5 marks]**

(g) Describe a distributed algorithm for deadlock detection. Is your solution able to detect only the *possibility* of a deadlock situation or can it guarantee to determine correctly whether a deadlock has occurred?

QUESTION 5(g)	**[3 marks]**

QUESTION 6 [25 marks]

(a) Show, with the aid of an appropriate diagram, how MPI can be used to implement Ricart and Agrawala's (fully) distributed algorithm for distributed mutual exclusion. Your description must include the various MPI calls (including the appropriate parameters) and the tags you use, but you don't need to write a complete program – just demonstrate that you can write the algorithm using MPI. You may assume you have access to a call `global_time()` which returns a unique global timestamp.

QUESTION 6(a)	**[10 marks]**

(b) Which OSI levels need to be implemented in a network router?

QUESTION 6(b)	**[2 marks]**

(c) What is CSMA/CD? Explain.

QUESTION 6(c)	**[5 marks]**

(d) What is the main advantage of a token ring architecture?

QUESTION 6(d)	**[3 marks]**

(e) Give a necessary and sufficient condition for serializability of two transactions.

QUESTION 6(e) **[3 marks]**

(f) In which states of a two-phase-commitment protocol are crashes of individual communication partners particularly hard to handle? Explain.

QUESTION 6(f) **[2 marks]**

Additional answers. Clearly indicate the corresponding question and part.

Additional answers. Clearly indicate the corresponding question and part.