DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

6.033—Computer System Engineering Handout 10—March 6, 1998

Quiz 1

This quiz contains 3 problems whose relative weights are given below. Please write your answers in the space provided, and hand in this booklet at the end of the examination.

Put your name on this cover sheet AND at the bottom of each page of this booklet. Be sure that you don't put part of the answer to a problem on the back of a sheet for another problem.

Some parts of some problems may be much harder than others. Read them all through first and attack them in the order that allows you to make the most progress. Some problems are intentionally ambiguous; be sure to write down any assumptions you make. Show your work, or you risk losing partial credit. Be neat. If we can't figure out your answer we can't give you credit. On answers that involve numbers, be sure to clearly specify the units of your answer.

Remember, the quiz is ***open book***.

Circle your recitation:

10:00	1. Gifford/Candea	13. Chapin/Almeida							
11:00	2. Gifford/Almeida	7. Chapin/Candea	12. Karger/Fu						
12:00	6. Rinard/Rusnak								
1:00	3. Rivest/Fu	8. Saltzer/Mazieres	9. Rinard/Sapuntzakis	11. Karger/Kwon	14. Ward/Rusnak				
2:00	4. Rivest/Sapuntzakis	5. Ward/Mazieres	10. Saltzer/Kwon						

For Official Use Only

Problem	Your Score	Maximum Score
1		30
2		40
3		30
Total		100

Name: _____

1. (*30 points*) ***Circle*** exactly one answer to the following questions (choose the best answer):

a. Recitations in 6.033:

1) Are not scheduled by the registrar, and require you to call Prof. Kaashoek at home to find a suitable time.
2) Are an essential part of the 6.033 educational experience, and participation in recitation discussions contributes to your final grade.
3) Are no longer held, and by not attending them you have cleverly saved a great deal of time waiting in empty rooms.
4) Have been recently increased in length from 50 minutes to 3 hours due to popular demand.

b. The errors in the Therac-25 system:

1) Were in the terminal handler, and that is why the company suggested removing the terminal up-arrow key.
2) Were dependent on timing.
3) Were caused by writing the system in assembly language which caused compatibility problems when the code was ported from the Therac-20 system.
4) Could have been fixed by correctly decoding the turntable position signals.

c. The process primitives of the UNIX system:

1) Create a new "forked" process and initialize its address space to zero.
2) Do not permit processes to wait because processes are easily terminated and new ones started.
3) Are used by the shell to create a single process for the command "ls | more" .
4) Let newly created processes inherit pipes to facilitate inter-process communication.

d. The Eraser system:

1) Detects synchronization errors by computing the "happens before" relation.
2) During compilation places checks at lock and unlock commands to log what data is being protected by locks.
3) Considers a variable "shared-modified" as soon as the first store to it executes.
4) May miss synchronization errors.

Name: _____

e. In Birrell's RPC system:

1) A remote procedure call will always be executed completely, exactly once, or not at all.
2) A remote procedure call will always be executed at least once, and may be executed more than once.
3) A remote procedure call will always be executed at most once, which includes the chance that the call may not be executed at all or may be partially executed.
4) In the absence of failures it is impossible for the client to know if a remote procedure call was executed or not.

Name: _____

2. *(40 points)* Ben Bitdiddle is called in to consult for Microhard. Bill Doors, the CEO, has set up an application to control the Justice department in Washington, D.C. The client running on the TNT operating system makes RPC calls from Seattle to the server running in Washington, D.C. The server also runs on TNT (surprise!). Each RPC call instructs the Justice department on how to behave; the response acknowledges the request but contains no data (the Justice department always complies with requests from Microhard). Bill Doors, however, is unhappy with the number of requests that he can send to the Justice department. He therefore wants to improve TNT's communication facilities.

   Ben Bitdiddle observes that the Microhard application runs in a single thread and uses RPC. He also notices that the link between Seattle and Washington, D.C. is reliable. He then proposes that Microhard enhance TNT with a new communication primitive, pipe calls.

   Like RPCs, pipe calls initiate remote computation on the server. Unlike RPCs, however, pipe calls return immediately to the caller and execute asynchronously on the server. TNT packs multiple pipe calls into request messages that are 1000 bytes long. TNT sends the request message to the server as soon as one of the following two conditions becomes true: 1) the message is full, or 2) the message contains at least 1 pipe call and it has been 1 second since the client last performed a pipe call. Pipe calls have no acknowledgements. Pipe calls are not synchronized with respect to RPC calls.

   Ben quickly settles down to work and measures the network traffic between Seattle and Washington. Here is what he observes:

One-way Seattle to Washington, D.C. latency:	12.5×10^{-3} seconds
One-way Washington, D.C to Seattle latency:	12.5×10^{-3} seconds
Channel bandwidth in each direction:	1.5×10^{6} bits / second
RPC or Pipe data per call:	10 bytes
Network overhead per message	40 bytes
Size of RPC request message (per call)	50 bytes = 10 bytes data + 40 bytes overhead
Size of pipe request message:	1000 bytes (96 pipe calls per message)
Size of RPC reply message (no data):	50 bytes
Client computation time between requests:	100×10^{-6} seconds
Server computation time per request:	50×10^{-6} seconds

   Note that the Microhard application is the only one sending packets on the link.

   Please show your calculations and put your final answer in the box.

   a. What is the transmission delay the client thread observes to transmit an RPC request message (the time required to transmit an entire message at the data rate of the link)?

   Answer: [ ]

Name: _____

b. Assuming that only RPCs are used for remote requests, what is the maximum number of RPCs per second that will be executed by this application?

Answer:

c. Assuming that all RPC calls are changed to pipe calls, what is the maximum number of pipe calls per second that will be executed by this application?

Answer:

d. Assuming that every pipe call includes a serial number argument, and serial numbers increase by one with every pipe call, how could you know the last pipe call was executed? (*Circle the best answer*)

1) Ensure that serial numbers are synchronized to the time of day clock, and wait at the client until the time of the last serial number.
2) Call an RPC both before and after the pipe call, and wait for both calls to return.
3) Call an RPC passing as an argument the serial number that was sent on the last pipe call, and design the remote procedure called to not return until a pipe call with a given serial number had been processed.
4) Stop making pipe calls for twice the maximum network delay, and reset the serial number counter to zero.

Name: _____

3. *(30 points)* Ben Bitdiddle has just developed a $15 single chip Network Computer - NC - and plans to scoop Intel and create a revolution in computing. In the NC network system the network interface thread calls the procedure Packet_Arrived when a packet arrives. The procedure Wait_For_Packet can be called by a thread to wait for a packet.

Part of the code in the NC is as follows:

```
VAR m: Thread.Mutex;                                  1
VAR packet_here: BOOLEAN;                             2
VAR Packet_Present: Thread.Condition;                 3
                                                      4
Packet_Arrived: PROCEDURE ();                         5
   BEGIN                                              6
      packet_here := TRUE;                            7
      Thread.Broadcast(Packet_Arrived);               8
   END;                                               9
                                                      10
Wait_For_Packet: PROCEDURE ();                        11
   BEGIN                                              12
      LOCK m DO                                       13
         WHILE NOT packet_here DO                     14
            Thread.Wait(m, Packet_Arrived);           15
         END;                                         16
      END;                                            17
   END;                                               18
```

a. It is possible that Wait_For_Packet will wait forever even if a packet arrives while it is spinning in the WHILE loop. Give an execution ordering of the above statements that would cause this problem. Your answer should be a simple list such as 1, 2, 3, 4.

Answer:

Name: _____

b. Write new version(s) of Packet_Arrived and/or Wait_For_Packet to fix this problem.

THE END

Name: _____