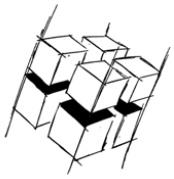
Exercise 6
Net Centric Systems
Summer Term 2015



Lena Després, M.Eng.
Steffen Schnitzer, M.Sc.

Multimedia Kommunikation (KOM)
Institut für Datentechnik
Fachbereich Elektrotechnik und Informationstechnik
Fachbereich Informatik (Zweitmitgliedschaft)

Rundeturmstraße 10
64283 Darmstadt

Telefon (06151) 16 - 6112
Telefax (06151) 16 - 6152
E-Mail ncs(at)kom.tu-darmstadt.de
Web http://www.kom.tu-darmstadt.de/

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Due Date:	06.07.2015 09:00
Published on:	22.06.2015
Version:	1.0
Examination:	
Overall Points:	122

Task 6.1 : RMI / RPC (14 points)

6.1.1: What are the main differences between Remote Method Invocation (RMI) and Remote Procedure Calls (RPC)? (2 points)

6.1.2: Assume you are programming a fairly complex server application in Java. Would you prefer to use RMI or RPC? Briefly justify your answer. (2 points)

6.1.3: Assume that you are using RMI and you need to decide if you want to allow access to an object via pass-by-value or pass-by-reference. The change for the object is a variable which is looped over an indefinite number of times. You estimate, that on average both methods are equally fast. Which reasons could there be to implement pass-by-value? In which case would you use pass-by-reference? (4 points)

6.1.4: Which failure semantics has an RPC mechanism to support the following use cases? Please, justify your answer briefly.
a) Bank transfer
b) Periodic mail check
c) One time weather forecast query (6 points)

Task 6.2: Synchronous and asynchronous message passing (15 points)

6.2.1: In the scenario in Figure 6.2.1 the processes communicate with each other. In the given pseudo code "send(d,v)" means the variable v is sent to destination d. Where sends and receives come without destination or source, assume a first-in-first-out pipe (queue) where sends are written into and receives read from (One message can only be received once).

Specify all possible outcomes(i.e. the printed outputs of process 1 and 2) …

a) assuming synchronous message passing.

b) assuming blocking receive messages and non-blocking send messages. (12 points)

Process 1	Process 2	Process 3

int i = 1;
int j = 2;
 	
send(p3, i);
send(p3, j);
 	
receive(i);
 	
print("First solution is: %d\n", i);
	int x = 10;
	int y = 20;
	
	send(p3, x);
	send(p3, y);
	
	receive(x);
	
	print("Second solution is: %d\n", x);
		int c = 0;
		int d = 0;
		
		receive(c);
		receive(d);
		send(c+d);
		
		receive(c);
		receive(d);
		send(c+d);

Figure 6.2.1: synchronous message passing

6.2.2: Which different asynchronous message passing variants exist and how do they differ with respect to buffering? (3 points)

Task 6.3 : Clocks (50 points)

6.3.1: In Cristian's Clock Synchronization Algorithm the interrupt handling time is needed. How is its length determined? Please write down the correct equation. What would happen if you leave out the interrupt handler time in the equation? (4 points)

6.3.2: Consider the application of Cristian's Clock Synchronization Algorithm in a setting where the maximum drift rate is at 10^{-5} and the clocks are never more than 100ms apart. The time it takes for the slave from inquiring to receiving an answer takes 10ms and the interrupt handling time at the master is 1ms. At the time the master reads its clock it is exactly $C_M(T_1) = 12:00$ and the slave's time (C_S) is earlier than the masters ($C_S=C_M-x \mid x>0$).

a) To what time is the slave's clock set after receiving the answer? ($C_S=C_M'(T_1)$)

b) When did the slave send the inquiry? (in time of C_M)

c) At which frequency does the client send the inquiry (at least)?

d) Which problem would occur if the slave's time was later than its master's ($C_S=C_M+x \mid x>0$) and how can this problem be solved? (6 points)

6.3.3: Briefly describe how Lamport's Clock Synchronization Algorithm works by comparing it to Cristian's Clock Synchronization Algorithm. (4 points)

6.3.4: In Figure 6.3.4 Lamport's Clock is depicted.
a) Name the events that cause event b3 (b3 is casually dependent on).
b) Name the events that are caused by b3 (are causally dependent on b3).
c) Name the events that are concurrent to b3.
d) Fill in the timestamps generated by Lamport's Clock (12 points)

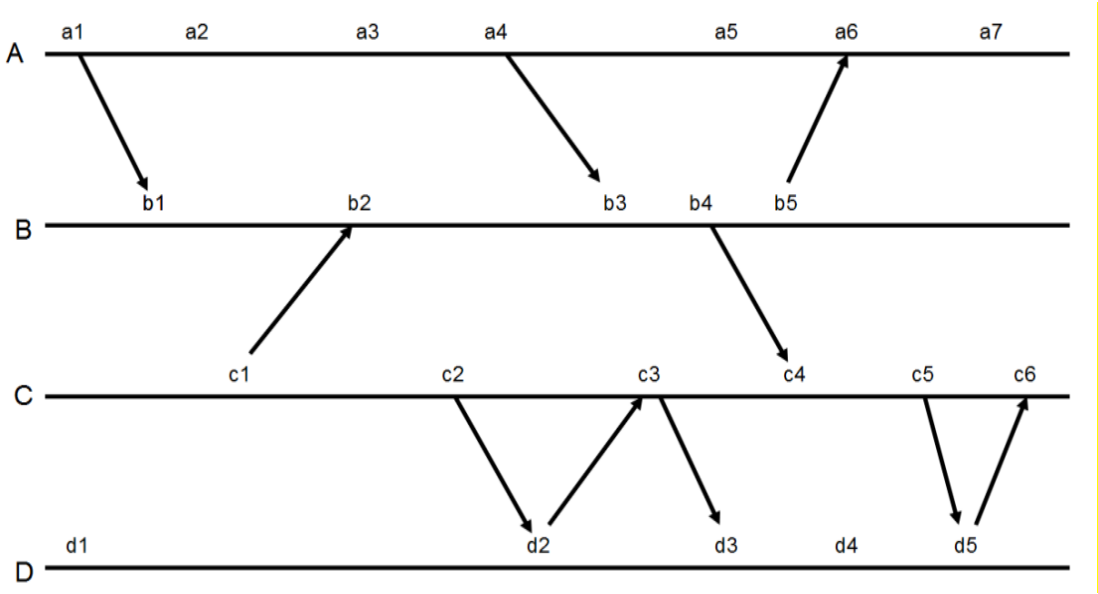


Figure 6.3.4: Lamport's Clock

6.3.5: Briefly describe the differences between logical and physical clocks. Name at least two differences. (2 points)

6.3.6: In which scenario would you prefer the usage of a physical clock? In which situation would you rather use a logical clock? Justify your decision briefly. (2 points)

6.3.7: How can the terms correct, accurate and synchronized be defined in the context of physical clocks? (3 points)

6.3.8: In Figure 6.3.8 a vector clock is depicted just as in 6.3.4. Specify the values of the vector for each of the events. (8 points)

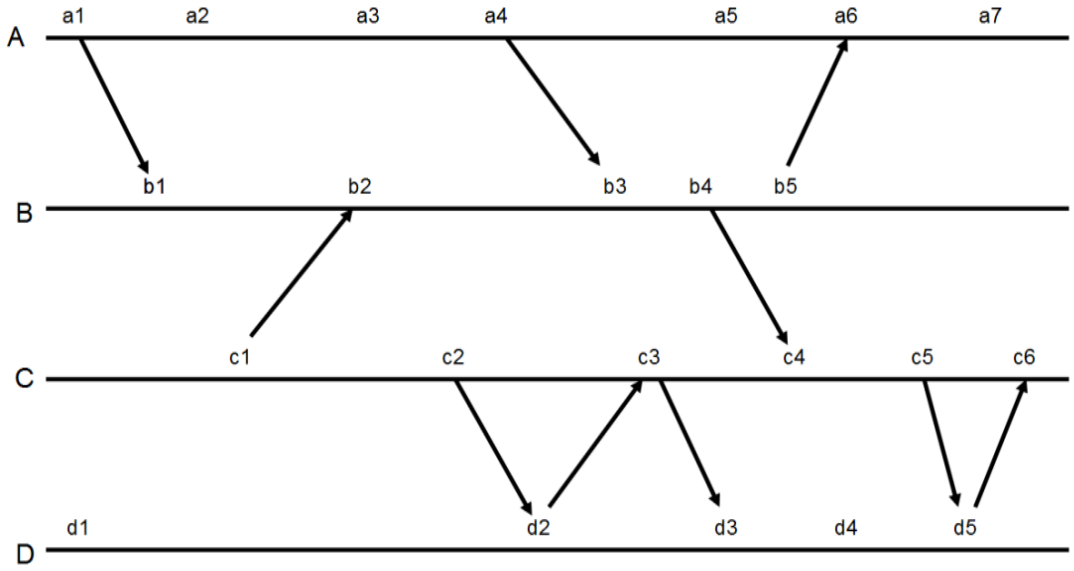


Figure 6.3.8: Vector Clock

6.3.9 In a vector clock, which event is concurrent to the given event (4,8,2)?

a) (2,1,6)

b) (3,8,4)

c) (2,8,4)

d) (2,8,2)

(4 points)

6.3.10: In a chat program users can send messages to a single user and receive messages that have been sent. Assume that Alex, Bob, Chris and Dave try to agree on a date to meet each other. Alex proposes several possibilities for dates and sends a message to the others telling them to send him their decision. Bob and Chris agree on Wednesday, but later Bob realizes that Thursday was fine with Chris too and fits him better, so he agrees on Thursday with Dave and goes offline. When Alex gets back and asks Chris and Dave on the agreed date he gets different answers. Which clock would you use to solve this problem? (5 points)

Task 6.4 : Java RMI Program (39 Points)

6.4.1: In this task you are supposed to hand in working code. You are provided with some sample setup code where you will be able to extend and correct the code in order to solve the task. Please provide besides the code some meaningful documentation by commenting and describing your submission.

Scenario:

The provided system is supposed to work for transferring money from and to bank accounts. The Client sends transferring orders to the TransferServer which processes the orders. The Client is able to create accounts and to put orders for transferring money from or to the account.

The functionality of the TransferServer is defined by the BankAccountTransferInterface.

For the first task ignore the classes of BankAccountInterface and BankAccountInterfaceImpl.

a) Correct and extend the classes of Client and BankAccount so far, that the pass-by-value methodology can be executed and the following scenario is followed:

The client creates a bank account with the starting amount of 100 and tries to subtract 200 directly afterwards, what should fail because the bank accounts cannot go below 0. Then it tries successfully to add 1 (as already implemented).

(6 points)

b) Adding 1 using the pass-by-value methodology did not work as expected. What did go wrong? (3 points)

c) Now try to implement the same functionality in the pass-by-reference methodology by using the BankAccountInterface and the BankAccountInterfaceImpl. Implement the same scenario as above. Some of the needed code already exists but only as comments. Add printouts to the already existing ones at the client to show how your method works.

(23 points)

d) Can you correct the code, so that the pass-by-value method works as well? If yes, do so and add outputs at the client.

(7 points)

Task 6.5: Connection-oriented and connectionless protocols (4 points)

6.5.1: TCP is a connection-oriented protocol that relies on IP which is connectionless. Briefly explain why this is possible. (2 points)

6.5.2: Which information is needed to identify a TCP socket? Which information is needed to identify a UDP socket? Why do they differ? (2 points)