UMEÅ UNIVERSITET
Institutionen för Datavetenskap

UMEÅ UNIVERISTY
Dept of Computing Science
Unique code: 1

EXAMINATION

Course: **5DV020/Distributed Systems**

Teacher in charge: Yvonne Löwstedt/Lars Larsson

Semester: HT-10
Date: 2010-11-02
Time: 09:00–13:00

Name: _____

Personal ID number: _____

Unique code for this examination: **1**

Note!

This examination will be graded anonymously. This sheet will be removed before the teacher receives the rest of the examination. The above code must therefore be on all other pages when you submit the examination to the examination supervisory staff. **Memorize** your code since it will be used as reference when the results are published.
Furthermore,

- Write the answers on the answers on the same paper as the question (the back of the paper may also be used).
- Mark the questions you have solved with a cross on the next page.
- The solutions should be neatly written. The train of thought should be easy to follow. All non-obvious assumptions must be explicitly stated.

Till skrivningsbevakaren: Avskilj detta försättsblad och stoppa i kuvert som skickas till Yvonne Löwstedt, Datavetenskap.

UMEÅ UNIVERSITET
Institutionen för Datavetenskap

UMEÅ UNIVERISTY
Dept of Computing Science
Unique code: 1

EXAMINATION

Course: **5DV020/Distributed Systems**

Teacher in charge: Yvonne Löwstedt/Lars Larsson

Semester: HT-10

Date: 2010-11-02

Time: 09:00–13:00

Unique code for this examination: **1**

Problem	**Solved**	**Points**
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
Sum		
Grade		

Question 1 (1 + 1 + 2 + 4 + 2 points)

Regarding message ordering, the course book[1] mentions something very interesting. If a system uses a sequencer for total ordering, and the messages sent by each process are FIFO-ordered, then the result is that all communication in the group is *both* causally *and* totally ordered. The book leaves the proof as an exercise for the reader. That means you. You're the reader. But first, let's warm up with some more general questions.

a. What is the purpose of using message ordering(s) in the first place?

b. Why is not Total-Causal always the best message ordering?

c. What advantage does the combined Total-Causal message ordering have over the regular Causal and Total orderings?

d. Show why, when using a sequencer, the above statement from the book about sequencer-supported Total-FIFO and Total-Causal holds.

e. Does the statement hold when you implement total ordering using something else, e.g. the ISIS voting algorithm? Why or why not? Motivate your response clearly.

Note that we do not require a formally correct proof, just show (very clearly!) why this has to be the case.

[1] *Distributed Systems: Concepts and Designs*, Colouris, Dollimore, Kindberg, 2005

This is a quite hard question, and you might need extra space to answer it fully.

Have another page, it's on us!

Question 2 (2 + 4 points)

Transactions is a very important concept in distributed systems — in particular ones that deal with databases of various kinds. The idea is to define not mere operations, but operation "groups" that can be applied to a distributed object. In that context, ACID is an acronym that is well-known and used in distributed systems as well as in databases. It describes properties that one requires from a transaction-aware system for the system to be useful.

a. What does ACID stand for? Name the four properties that make up the acronym, and give a brief description of them in the context of distributed systems.

b. There are some problems related to distributed objects that can be accessed using transactions. Three classical problems are the *dirty read*, *lost update*, and *inconsistent retrievals* problems. Describe **two** of these (you may choose which two of the three), and give simple and clear examples that illustrate what goes wrong.

Question 3 (4x2 points)

Cryptography is used in many different security mechanisms, but sometimes in a different way. Please briefly explain the following concepts and mention when they are used:

a. A private key

b. Public key infrastructure

c. A digital signature

d. A ticket

Question 4 (3 + 1 points)

We discussed both internal and external time synchronization algorithms during the course, and a few algorithms that implement either internal or external synchronization. The Berkeley algorithm is used for internal synchronization.

a. Draw a figure of how the system is set up, including which messages are sent between processes, and

b. discuss upper and lower bounds on the accuracy this system can give you, given the contents of the message exchange.

Question 5 (4 points)

A naive way of implementing something similar to an election is to order all processes in a list and always state that the leader is the first non-crashed member in the list. Explain (note the amount of points we're awarding this question) clearly why this is *not* a correct election algorithm according to the requirements that hold for all reasonable election algorithms.

Question 6 (1 + 4 + 3 points)

During the lectures, we have discussed several algorithms for mutual exclusion in distributed systems. Show us what you remember!

a. What is mutual exclusion used for?

b. Choose **one** of the following three algorithms that we covered: ring-based, Ricart and Agrawala's algorithm using timestamps, or Maekawa's voting algorithm and draw a figure of how the nodes communicate.

c. Given your choice, clearly state whether it has the following properties: *liveness*, *safety*, and *happened-before ordering*. Motivate your response clearly.

This page intentionally left blank(-ish).

Question 7 (1 + 3 + 3 points)

Handling transactions in distributed systems is tricky business.

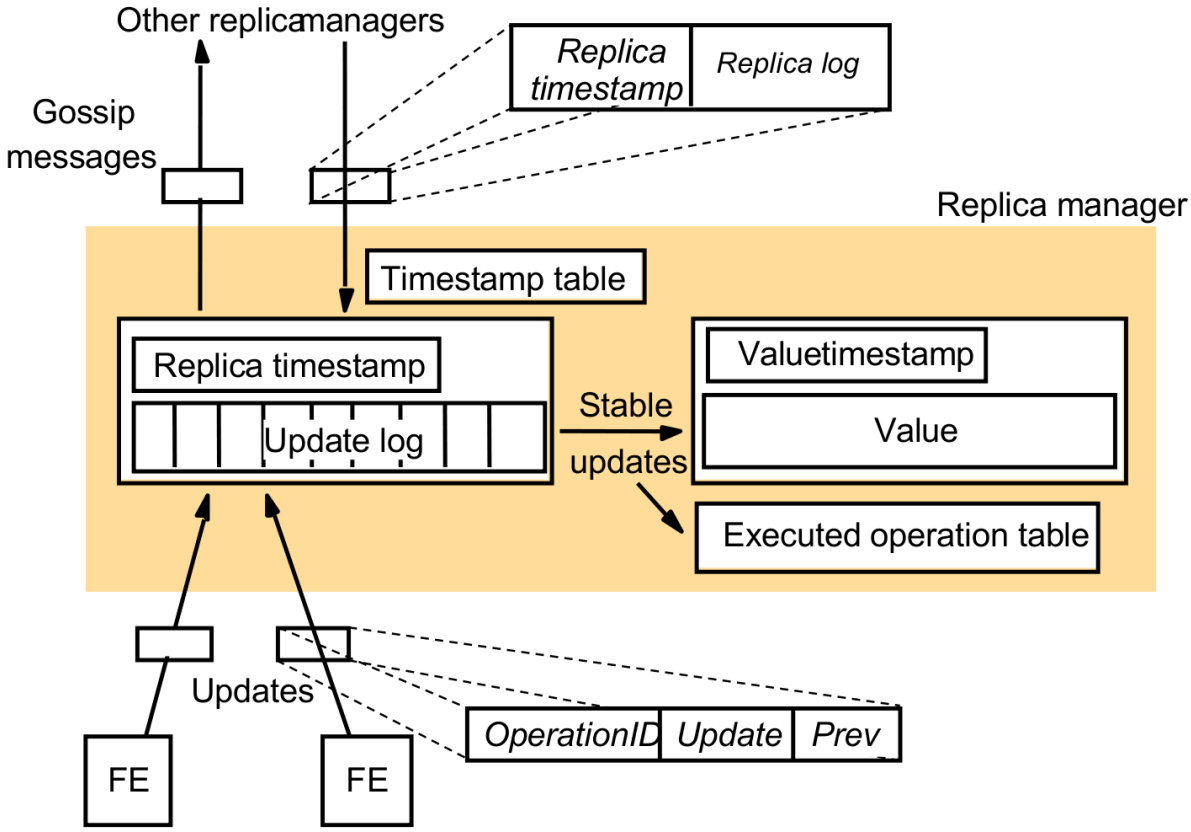
a. What are atomic commit protocols used for and why do we need them?

b. Two-phase commit is a well-known atomic commit protocol. Please illustrate how it works for a system with three nodes and one coordinator in it.

c. The two-phase commit protocol is quite safe against errors. How are crashing (and later restarting) processes handled according to the protocol? Keep in mind that either the participants or the coordinator could crash/restart.

Question 8 (6 points)

Here's a picture showing the details a component in the Gossip architecture.



In this question, we're not interested in the *details* but in the bigger picture (we're giving you the details right up there for free, after all!). Please explain *the main benefits of using Gossip* and what the protocol/system/architecture is used for and what it offers compared to other replication schemes.

Consider this an essay question, so provide us with clear and detailed but still concise answer.

Have another page!

Question 9 (2 + 2 points)

The following is an excerpt from the security policy of Murdoch University in Australia (used without permission):

1. Your Student Network account is provided by the University in your name for your use only.

2. You must not share your account with family, friends or make your password available to any other person.

3. You should change your password at least every 30 days.

4. You may not use the account of any other person. If you inadvertently gain such access to any unauthorized information, you should advise Helpdesk staff immediately.

5. In certain circumstances you may share an account with others where shared duties apply. Such accounts will be specifically authorized by the Director ITS or delegate. In such cases all sharers are jointly responsible for the account but may not share with others outside the group.

6. You MUST NOT attempt to find the password of another user or access their account in an unauthorized user name.

a. What is the purpose of a security policy?

b. Reason about *which* of the above bullets can be enforced by mechanisms and *how*.

New page!

Question 10 (-3 to 3 points)

The following questions require only a true or false answer. Correct answers give 0.5 points, whereas incorrect answers are penalized with -0.5 points. Note that the total from the question may be negative, and this will impact your final score. No answer is the safest option, and counts as 0 points. Any text besides "true" or "false" will not be taken into consideration.

BitTorrent is suitable for transporting streaming live video	
The instantaneous difference between the readings of any two clocks is called clock drift	
Forward and backward validation of transactions can be combined, turning it into a pessimistic concurrency control scheme	
TLS and SSL can interoperate	
Java RMI offers *exactly once* invocation semantics	
NTP symmetric mode exchanges messages similarly to Cristian's algorithm	