TDA596 / DIT240 (2nd academic period 2015/2016)

Exam: Distributed Systems

12. Jan. 2016

Examiner: Olaf Landsiedel

Contact: olafl@chalmers.se, office phone: 772 10 96 or 070-970 24 43

Means allowed: Nothing except paper, pencil, pen and English - xx dictionary.

Please answer questions 1 to 6

General information: All questions should be answered in English. Write clearly and use the pages in a structured way so your answers are easy to read. Each question answer should be started on a new sheet of paper. All answers should be motivated, explained, elaborated, detailed, precise and accurate.

Important suggestion: Read all questions before answering. Plan your time so that you can (at least) write a brief answer to all questions (and sub-questions). Please notice the weight that is given to each question (and sub-question).

Grading: GU: G 24p, VG 48p ; CTH: 3:a 24p, 4:a 36p, 5:a 48p of maximum 60 points.

Review: Please keep your exam code. Information about individual exam review will be published on the course website.

Department of Computer Science and Engineering
Chalmers University of Technology

CHALMERS

1. Basics about Distributed Systems (10 points)

1 a) (1 points) Define the term “Distributed System”. Be brief and precise.

1 b) (1 points) The following statement is attributed to Thomas J. Watson, Chairman and CEO of International Business Machines (IBM), in 1943: “I think there is a world market for maybe five computers”. Please state the key consequences for distributed systems if this sentence had been the correct vision.

1 c) (2 points) When designing or developing a distributed system, its distributed nature presents a number of challenges, which we discussed in the course. List at least six such challenges.

1 d) (4 points) We discussed the Dining Philosophers Problem: N silent philosophers sit at a round table with bowls of spaghetti. Forks are placed between each pair of adjacent philosophers. Each philosopher must alternately think and eat. However, a philosopher can only eat spaghetti when he has both left and right forks. Only one philosopher can hold each fork and so a philosopher can use the fork only if no other philosopher is using it. After he finishes eating, he needs to put down both forks so they become available to others. A philosopher can take the fork on his right or the one on his left as they become available, but cannot start eating before getting both of them. Note, eating is not limited by the amount of spaghetti left; an infinite supply is assumed. (Text adapted from Wikipedia).
Sketch in pseudo code a solution to this problem that is deadlock free, starvation free, and distributed (i.e., does not require a central entity for coordination). Briefly explain why your solution is deadlock free and starvation free.

1 e) (2 points) Ethical challenges: Certain distributed systems such as BitTorrent and TOR trigger ethical challenges. List and briefly discuss two ethical challenges for each of these two systems.

2. Mutual Exclusion and Election and Naming (10 points)

2 a) (1 points) Define the terms "Mutual Exclusion" and "Election", as used in the context of this course. Be brief and precise.

2 b) (3 points) Bully Algorithm for Leader Election.

- Please describe how the Bully Algorithm algorithm works.
- How does the Bully algorithm deal with nodes failing during election?
- What message complexity (please use the "O"-notation) does the algorithm have and why?

2 c) (3 points) In the course we discussed two concepts for name resolution: Iterative and recursive name resolution. Briefly describe each concept and highlight their key differences.

2 d) (3 points) In the lecture we discussed the concept CAN. CAN is a Distributed Hash Table (DHT). Answer the following questions about CAN:

- What topology do the nodes form?
- What operations does a DHT, e.g., CAN, provide?
- How is redundancy in CAN achieved?
- In CAN, how many hops does it take at most to lookup a data item? (Assume that the number of nodes is "n", and dimensions are "d").
- How does a node join a CAN DHT?
- Neighbor Table: Which nodes are stored in a neighbor table of a node in the DHT?

3. Time and Synchronization (10 points)

3 a) (1 points) In the course we discussed the concept of clock synchronization for physical clocks. Why is clock synchronization important in Distributed Systems that rely on physical clocks?

3 b) (3 points) Assume we need to synchronize the physical clocks of "n" nodes but do **not** have access to a reference clock.

- Which algorithm would you choose and why?
- Briefly illustrate that algorithm. You can draw a figure to support your argumentation.

3 c) (3 points) Assume we need to synchronize the physical clocks of "n" nodes to an accurate reference clock.

- Which two algorithms did we discuss in the course that could you choose and why?
- Briefly illustrate one of the two algorithms. You can draw a figure to support your argumentation.

3 d) (3 points) In the course, we discussed how Vector Clocks help to distinguish causally related events and concurrent events.

- Please explain how this can be achieved
- Below you see pairs of Vector clocks. Note for each pair whether they denote concurrent or causally related events. Briefly explain your reasoning
    - Are these two events causally related or concurrent?

Event on Node 1	Event on Node 2
VC[1]= 5	VC[1]= 5
VC[2]= 2	VC[2]= 8

    - Are these two events causally related or concurrent?

Event on Node 1	Event on Node 2
VC[1]= 1	VC[1]= 0
VC[2]= 0	VC[2]= 6

4. Consistency and Replication (10 points)

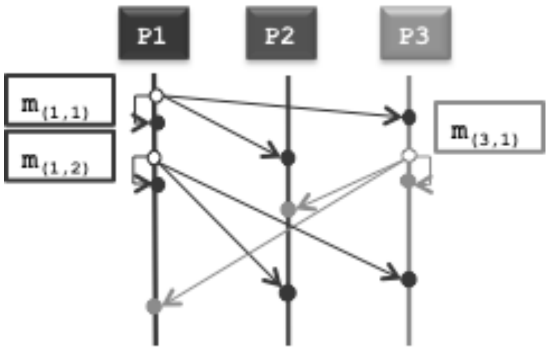
4 a) (1.5 points) In the lectures we discussed the concepts of "Replication", "Consistency" and "Eventual Consistency".

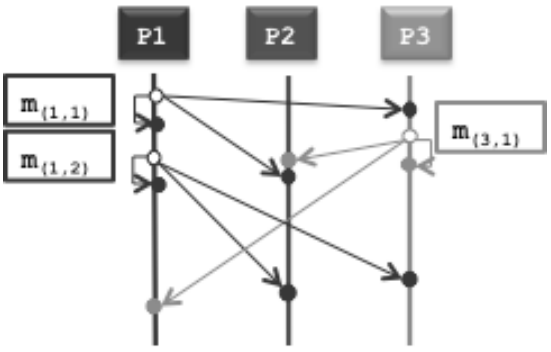
i. Please define the term "Replication"
ii. Please define the term "Consistency"
iii. Please define the term "Eventual Consistency"

4 b) (1.5 points) List three benefits that replication provides. Explain these benefits briefly.

4 c) (4 points) We discussed the concepts of Total Ordering, Sequential Ordering, and Causal Ordering

- Briefly explain each concept.
- Below you see two figures. For each figure, please note weather it describes Total Ordering, Sequential Ordering, and Causal Ordering. Briefly describe your decisions.





4 d) (3 points) We discussed the concept of a Centralized Active Replication Protocol.

- Briefly explain this concept. You can draw a figure to support your argumentation.
- Active replication uses a central entity for parts of its operations. Briefly explain why this is a reasonable design.

5. Fault Tolerance (10 points)

5 a) (2 points) Failure Models: In the lecture we discussed different failure models. Please note four of them and describe each briefly.

5 b) (4 points) We discussed the "Byzantine Generals Problem".

- In the "Byzantine Generals Problem" there are honest generals and dishonest generals (traitors). What is the goal of the honest generals? What is the goal of the traitors?
- In the lecture we introduced an algorithm with multiple phases to enable consensus among the generals. Explain the algorithm and its different phases.
- Under what conditions can the generals achieve consensus. How many honest generals are required, assuming that there are *k* dishonest ones?

5 c) (4 points) We discussed the "Two Phase Commit" Protocol. As the name states, it consists of two phases.

- Please name and describe phase 1 briefly.
- Please name and describe phase 2 briefly.
- Please discuss what happens in case of a failure during *phase 1*, i.e., a node not replying because it crashed.
- Please discuss what happens in case of a failure during *phase 2*, i.e., a node not replying because it crashed.

6. Applications

6 a) (5 points) We discussed TOR, which enables, for example, anonymous Internet browsing.

- Briefly explain how TOR provides anonymous Internet browsing. You can draw a figure to illustrate your argumentation.
- TOR also allows so called hidden services. Please briefly explain what a hidden service is and how TOR enables it. You can draw a figure to illustrate your argumentation.

6 b) (5 points) Describe the MapReduce algorithm. Split it into its phases. For each phase include: what it does and who is responsible (the MapReduce framework or the programmer).