Introduction to Distributed Systems
WT 19/20

Assignment 1

Submission Deadline: Monday, 04.11.2019, 08:00

- *Submit the solution in PDF via Ilias (only one solution per team).*
- *Respect the submission guidelines (see Ilias).*

---

1 Transparency Levels [11 points]

Various transparency definitions were presented in the lecture. The transparency properties of a system should aid the user and the application programmer to perceive the system as a whole rather than a collection of independent components. How does transparency apply to the following scenarios?

a) [2 points] Consider a web service that returns electronic copies (MP3) of songs for a given title. We can invoke the servie through the web browser by using the following URL

`http://mysong.uni-stuttgart.de/songs?title=<your title>`

Is the access to songs provided by this web service location transparent? Justify your answer.

b) [2 points] The host of `mysong.uni-stuttgart.de` of the previous service is insufficient reliable, therefore we have a second web service with a copy of all songs. We access the second service by using the following URL

`http://mysong2.uni-stuttgart.de/songs?title=<your title>`

If the first service call does not work, we use the second service as a fallback. Is the access to songs provided by both services replication transparent? Justify your answer.

c) [2 points] You do not want to waste your time with testing if the web services work, hence you to write a message with your song title to Otto Bibartiu. Otto will query the song at some web service and reply the song file back to you.

Is the access of songs replication-transparent for you (i.e., the client)? Justify your answer.

d) In the following, we discuss a simple replication protocol using the write-all and read-one semantics. Assume there are no node and communication failures. The read and write operations are defined as follows:

- Write operation: the client writes a copy to all replicas (write-all). Before writing a copy to all replicas, an explicit lock needs to be acquired. The locks are released after all writes have been performed.
- Read operation: the client reads a copy from one replica (read-one). Before reading a copy, the explicit lock needs to be acquired again. Once a client has locked a copy, no other clients may access the copy. The lock is released after reading.

i. [2 points] This protocol ensures that each read operation delivers the most recent value written to any copy. Argue why.

ii. [3 points] Locking introduces overhead. Assume we have n copies. Design a protocol where only $n-1$ with $n \geq 3$ copies need to be locked for writing. How are read operations performed? Make sure that a minimal number of copies need to be locked. Make sure that the read delivers most recent write.

2 System Models [10 points]

Assume we perform a particle simulation of a physical experiment. The simulation is very large, therefore, we distribute the simulation across n processes $P_1, \dots P_n$, where each process simulates a smaller portion of the experiment. The simulation consists of multiple execution steps. After each execution step, all processes need to synchronize by publishing their results to all other processes. Once all processes have received all results, the next execution step begins.

This synchronization protocol works as follows: P1 begins by broadcasting its results in a message to each process. Then P2 sends its results, and so on, until finally P_n sends its results as shown in Figure 1. The local time of a process P_i at which its message m is broadcasted is denoted as $t_{P_i}^{\text{send}}(m)$. The local time when a process P_j receives a broadcast message m is denoted as $t_{P_j}^{\text{receive}}(m)$

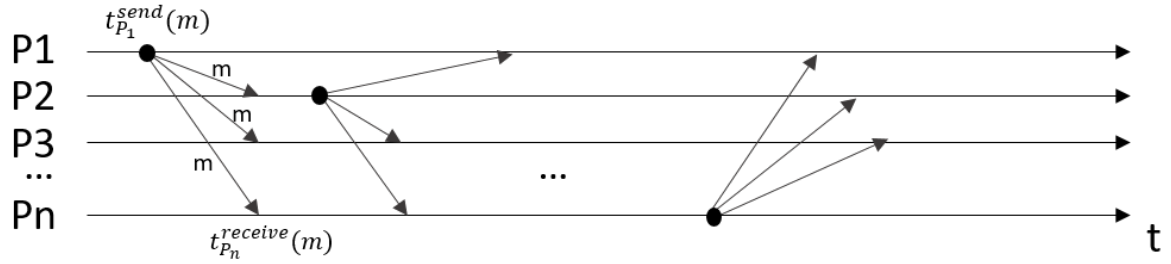

Figure 1: Time-diagram of the synchronization protocol.

a) [4 points] In order to start eventually with the next execution step, the synchronization protocol must fulfill the following condition: a process broadcasts a message only after all other processes received the previous message, i.e., all processes receive all messages in the same order. What assumptions do we need for our system model to guarantee this requirement?

b) Assume that the local send-time of P_i and the local receive-time of any other process P_j of the broadcast message m has the following property:

$$t_{P_j}^{\text{receive}}(m) - t_{P_i}^{\text{send}}(m) < \delta t$$

for some given δt within

$$d \leq \delta t \leq k,$$

where $k \geq 2d$.

i. [4 points] Define a protocol where each process sends it message as soon as it is certain that all other processes have received the previous broadcast message. Thus, fulfilling the requirement that all processes receive the messages in the same order. Justify your answer.

ii. [2 points] How long is the execution time of the synchronization protocol in worst-case? Justify your answer.

3 Three-Army-Problem [6 points]

In the following, we discuss a variation of the Two-Army-Problem presented in the lecture. Similar to the Two-Army-Problem, the goal of the Three-Army-Problem is to reach agreement among *three* divisions about the order and time of the attack. In this case, it must be guaranteed that the largest division attacks first, the second largest division attacks second, and the smallest division attacks last.

The protocol must be clearly divided into two phases:

1. The divisions coordinate the **order of attack** (by division size).
2. The *largest* division determines the **time of attack** and informs the other divisions.

Furthermore, attacks of subsequent divisions should take place **at least** k **minutes apart**.

a) [2 points] What are the properties of the underlying system that should be defined in the failure model?

b) [4 points] Assume a *synchronous system* with no failures and the clocks of all three divisions are perfectly synchronized (i.e. no drift, no clock skew). A messenger between any two divisions may take up to d minutes with $d < k$. You may assume that all divisions are of different size. An arbitrary division initiates the protocol at time t_0. Briefly describe a protocol for the Three-Army-Problem that fulfills the following two conditions:

   1. The time between the first attack and the last attack must be minimal.
   2. The first attack should start as soon as possible (without violating the first condition).

   Describe and illustrate your protocol through a time-space-diagram.

4 System Availability [6 points]

Consider a distributed system with two nodes X and Y. We measured for each node the following up-time synchronously as shown in Figure 2:

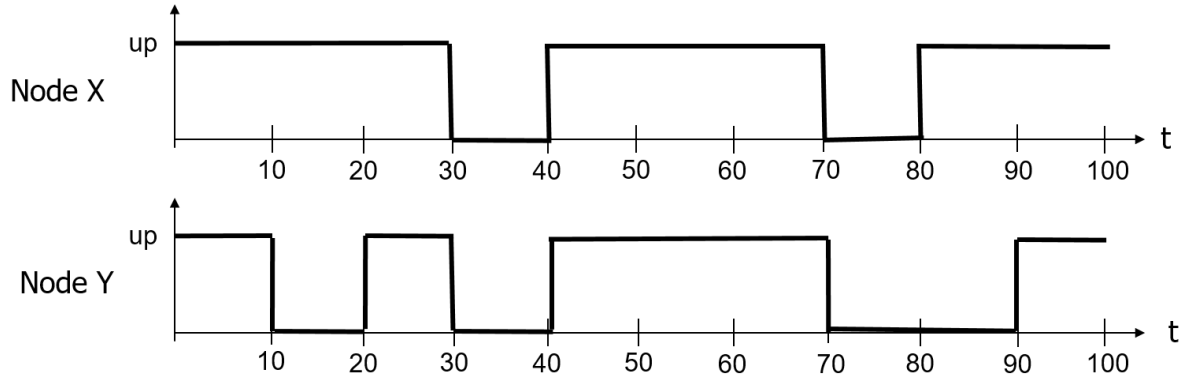

Figure 2: Up-time diagram of two nodes

a) [1 point] Compute the availabilities A_X, A_Y of both nodes for the depicted time frame.

b) [1 point] What is the availability of the distributed system, if at least one node is sufficient for a working system? *Note: assume the availability of the nodes is independent identical distributed (i.i.d).*

c) [2 points] Assume the availability of the nodes is not independent. What is the probability of observing node Y as up given that node X is up? What is the probability of observing node Y as up given that node X is not working? What conclusions do you draw about the failure dependence between node X and Y?

d) [2 points] Assume the availability of the nodes is not independent. Again, what is the availability of the distributed system, if at least one node is sufficient for a working system? Compare your result with previous result from question 4.*b*.