Introduction to Distributed Systems
WT 20/21

Assignment 1

Submission Deadline: Monday, 23.11.2020, 10:00

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

---

1 Transparency Levels [12 points]

In the lecture, you learned various transparency definitions. 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) [3 points] Consider a web service that returns electronic copies (MP3) of songs for a given title. We can invoke the service through the web browser by using the following URL

`http://129.69.184.135/mysong.mp3`

What transparency levels are provided? Name at least three and justify them.

b) [5 points] Now, we use a different web service that supports the following access pattern:

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

What transparency levels are provided? Name at least five and justify them.

c) In the following, you will design a simple replication protocol to read and write documents from and to replicas. The replication protocol should guarantee the following property: a client application that executes the replication protocol to read a copy of the document should deliver the most recent copy written of all copies.

Each read or write operation needs to acquire an explicit lock of the copy before the client can access the copy. Assume there are no node and communication failures.

i. [2 points] Describe a replication protocol that needs to write (and lock) only a majority of the copies.

ii. [2 points] Argue why your protocol ensures that each read operation delivers the most recent document written to any copy.

2 System Models [12 points]

Assume a networked system with two processes P_1 and P_2, where process P_1 sends a request message m_{req} to the P_2, and P_2 sends the response message m_{res} to P_1 back after doing some processing. The local clocks of P_1 and P_2 may drift with a bounded drift rate greater than 0, i.e., the drift rate is greater than 0 but smaller than infinity. The real ("wallclock") time at which a message is sent is denoted as $t_{\text{real}}^{\text{send}}(m)$. The local time of a sending process P at which a message m is sent is denoted as $t_P^{\text{send}}(m)$. The local time when a process P receives a message m is denoted as $t_P^{\text{receive}}(m)$ Assume that messages (request and responses) are only sent at most once by P_1 and P_2, respectively. Moreover, P_1 and P_2 use a connectionless, unreliable communication protocol at the transport layer to exchange their messages.

a) [2 points] What are the necessary system assumptions to guarantee that P_1 eventually receives a response after sending a request? Justify your answer. *Note: Here and in the following, we do not expect concrete or even symbolic numbers. A qualitative description of the required assumptions is sufficient.*

b) [3 points] What assumptions are required about the system to guarantee the following property? P_1 receives a response latest at time $t_{real}^{\text{receive}}(m_{\text{res}}) = t_{real}^{\text{send}}(m_{\text{req}}) + t_{\max}$ for some $t_{\max}$ with a fixed positive value.

c) [3 points] What assumptions are required about the system to guarantee the following property? Justify your answer. $|t_{P_2}^{\text{receive}}(m_{\text{req}}) - t_{P_1}^{\text{send}}(m_{\text{req}})| < \delta t$ for some given δt with $0 \leq \delta t < \infty$.

d) [4 points] What assumptions are required about the system to guarantee the following property? Justify your answer. P_1 sends two request messages m_{req1} and m_{req2} at $t_{P_1}^{\text{send}}(m_{\text{req1}})$ and $t_{P_1}^{\text{send}}(m_{\text{req2}})$ with $0 < t_{P_1}^{\text{send}}(m_{\text{req1}}) < t_{P_1}^{\text{send}}(m_{\text{req2}})$
and eventually receives the respective responses m_{res1} and m_{res2} at
$t_{P_1}^{\text{receive}}(m_{\text{res1}})$ and $t_{P_1}^{\text{receive}}(m_{\text{res2}})$ with
$t_{P_1}^{\text{send}}(m_{\text{req2}}) < t_{P_1}^{\text{receive}}(m_{\text{res1}}) < t_{P_1}^{\text{receive}}(m_{\text{res2}})$.

3 Three-Army-Problem [6 points]

In the following, we discuss a variation of the Two-Army-Problem presented in the lecture. The Three-Army-Problem intends to reach an agreement among *three* divisions about the order and time of the attack. In this case, the largest division must attack first, the second-largest division must attack 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 seeks consensus with the other divisions.

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

a) [2 points] What are the minimal assumptions about the underlying system so that the problem can be solved?

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 sizes. An arbitrary division initiates the protocol at some 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 1:

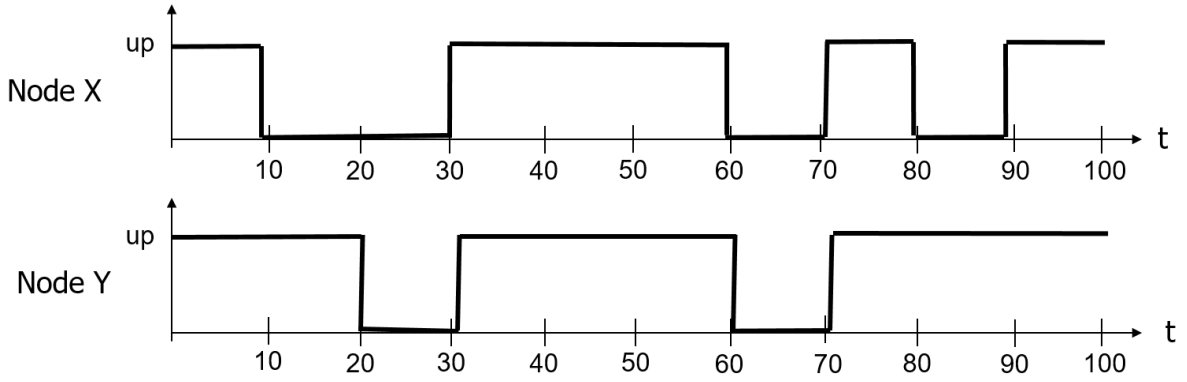

Figure 1: Up-time diagram of two nodes

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 distributed.*

c) [2 points] Assume the availability of the nodes is not independent. What is the probability of observing node X as up given that node Y is up? What is the probability of observing node X as up given that node Y is not working? Are node X and Y fault independent?

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*.