Albert Ludwigs University Freiburg
Department of Computer Science
Prof. Dr. Stefan Leue and Corina Apachiţe
Distributed Systems - WS 2001/2002

Assignment 3 - Solutions

Question 3.1 Discuss whether the following operations are *idempotent*:

- writing data to a file;
- appending data to a file.

It is a necessary condition for idempotence that the operation should not be associated with any state? (1 point)

Answer 3.1 An *idempotent operation* is an operation that can be performed repeatedly with the same effect as if it had been performed exactly once. (0.3p) If we want to discuss whether the writing operation is an idempotent operation, we should have a small discussion: how this operation is executed?

- if the write operation is applied at the write pointer, then the operation is no more idempotent (every application of it will change the state of the write-pointer).
- if the write operation is applied every tome at the same specified location, then the operation is idempotent, since it can be repeated with the same effect.

0.25p

The operation of appending data to a file is not idempoteny because the file is extended each timi this operation is executed. (0.25)

If we are looking at the definition of an idempotent operation, we can observe that the effect of an idempotent operation is independent of th eprevious operation. How can these effects be transmitted from operation to another? By means of states. So, the effects of an idempotent operation should not depend on any server state. (0.2p)

Question 3.2 A clock is reading 10:27:54.0 (hr:min:sec) when it is discovered to be 4 seconds fast. Explain why it is undesirable to set it back to the right time at that point and show (numerically) how it should be adjusted so as to be correct after 8 seconds has elapsed. (2 points)

Answer 3.2 We can not set back the clock because some application could timestamp events under the assumption that clocks always advance.

In our case we have the errant clock, let's call it E, and the hardware clock H (which is supposed to advance at a perfect rate). Let's now construct a

software clock such that after 8 seconds we can replace the errant clock with the software clock in good conditions. Let's denote the software clock with S, then: $S = c(E - T_{skew}) + T_{skew}$, where $T_{skew} = 10:27:54$ and c is to be found.

We know that $S = T_{skew} + 4$ when $E = T_{skew} + 8$, so: $T_{skew} + 4 = c(T_{skew} + 8 - T_{skew})$, and $c = 0.5$. We obtain the formula:

$$S = 0.5(E - T_{skew}) + T_{skew} \ (when \ T_{skew} \leq E \leq T_{skew} + 8).$$

(2 p)

Question 3.3 A client attempts to synchronize with a time server. It records the round-trip times and timestamps returned by the server in the table bellow. Which of these times should it use to set its clock? To what time should it set it? Estimate the accuracy of the setting with respect to the server's clock. If it is known that the time between sending and receiving a message in the system concerned is at least 8 ms, do your answers change?

Round-trip (ms)	Time (hr:min:sec)
22	10:54:23.674
25	10:54:25.450
20	10:54:28.342

(3 points)

Answer 3.3 The client should choose the minimum round-trip time of 20ms=0.02s. It then estimates the current time to be 10:54:28.342+0.02/2=10:53:28.352. The accuracy is +/-10ms. (1 p)

If the minimum message transfer time is known to be 8 ms, then the setting remains the same but the accuracy improves to +/-2ms. (1p)

Question 3.4 By considering a chain of zero or more messages connecting events e and e' and using induction, show that $e \rightarrow e' \Rightarrow L(e) < L(e')$. (3 points)

Answer 3.4 Induction basis:

- If e and e' are successive events occuring at the same process, then teh result is immediate from LC1. (1p)
- If there is a message m such that $e = send(m)$ and $e' = receive(m)$, then the result is immediate from LC2. (1p)

Inductive step: assume that the result to be proved is true for all pairs of events connected in a sequence of events of length N or less.Now assume that e and e'

are connected in a series of events $e_1, e_2, \ldots, e_N, e_{N+1}$ occuring at one or more processes such that $e = e_1$ and $e' = e_{N+1}$. Then $e \rightarrow e_N$ and so $L(e) < L(e')$ by the induction hypothesis. By LC1 and LC2 we have $L(e_N) < L(e')$. Therefore $L(e) < L(e')$. (1p)

Question 3.5 What do you understand under the notion of *protocol*? (1 point)

Answer 3.5 A protocol is a set of rules which have to be followed in the course of some activity. Originally, the term was used solely of human activities especially those of formal kind. The chef of protocol for a Head of State, for example, sets formal rules for how activities take place according to the niceties of diplomatic practice. But protocols must also be followed in less elevated spheres, such as games of all kinds, the way in which conversations are conducted, and in fact all activities which are governed by custom and conversation. If the protocol is not followed, the activity will not be successful.

Let's consider the communication protocol, and in particular those which regulate communication between computers. The characteristics of protocols mentioned above are equally evident in this case: a set of *formal rules*, that governs the exchange of information, and the communication activity *fails* if the protocol is not correctly followed.