CS 677 Homework 2

Due 03/15/2013 on moodle

1. Strong mobility in Unix systems could be supported by allowing a process to fork a child on a remote machine. Explain how this would work.
2. Consider the following *last-one* call semantics for RPCs: a caller repeatedly calls the server until a response is received.
    a. Give an example of a service for which this semantics is appropriate.
    b. Give an example of a service for which this semantics is NOT appropriate. Explain why and indicate the semantics that would be appropriate for your example.
3. Suppose you could make use of only transient synchronous communication primitives. How would you implement primitives for transient *asynchronous* communication?
4. Does it make sense to implement persistent asynchronous communication by means of RPCs?
5. RPC cannot support local references (such as pointers), as these refer to objects only locally accessible. Instead, global object references should be used, if possible. Outline an implementation of such a reference.
6. What is the main difference between a remote method invocation (RMI) and a RPC.
7. With persistent communication, a receiver generally has its own local buffer where messages can be stored when the receiver is not executing. To create such a buffer, we may need to specify its size. Give an argument why this is preferable, as well as one argument why this is not preferable.
8. In contrast to naming service like DNS, general-purpose directory services such as X.500 are much harder to scale. Why?
9. When a node synchronizes its clock to that of another node, it is generally a good idea to take previous measurements into account as well. Why? Also, give an example of how such past readings could be taken into account.
10. Use an example to show Lamport timestamps are not sufficient to capture causality.