

## **Tutorial 3**

### **Communication**

1. Why are the transport level communication services often inappropriate for building distributed applications?
2. A reliable multicast service allows a sender to reliably pass messages to a collection of receivers. Does such a service belong to a middleware layer, or should it be part of a lower-level layer?
3. Consider a procedure *incr* with two integer parameters. The procedure adds 1 to each parameter. Now suppose that it is called with the same variable twice, for example, as *incr(i,i)*. If *i* is initially 0, what value will it have afterward if call-by-reference is used? How about if copy/restore is used?
4. Assume a client calls an asynchronous RPC to a server, and subsequently waits until the server returns a result using another asynchronous RPC. Is this approach the same as letting the client execute a normal RPC?
5. Instead of letting a server register itself with a daemon as is done in DCE, we could also choose to always assign it the same endpoint. That endpoint can then be used in references to objects in the server's address space. What is the main drawback of this scheme?
6. Java and other languages support exceptions which are raised when an error occurs. How would you implement exceptions in RPCs (and RMIs)?
7. Suppose that you could make use of only transient asynchronous communication primitives, including only an asynchronous receive primitive. How would you implement primitives for transient *synchronous* communication?
8. Now suppose that you could make use of only transient synchronous communication primitives. How would you implement primitives for transient *asynchronous* communication?
9. Routing tables in IBM WebSphere MQ, and in many other message-queuing systems, are configured manually. Describe a simple way to do this automatically.
10. Does it make sense to implement persistent asynchronous communication by means of RPCs?
11. Explain why transient synchronous communication has inherent scalability problems, and how these could be solved?
12. Give an example showing that multicasting can be useful for discrete data streams.
13. Despite that multicasting is technically feasible, there is very little support to deploy it in the Internet. The answer to this problem is to be sought in down-to-earth business models: no one really knows how to make money out of multicasting. What scheme can you invent?