Dept. Math. & Comp. Sc. **Midterm Distributed Systems**
Vrije Universiteit **29.10.2001**

1a What is the difference between synchronous and isochronous transmission mode? For each mode, give an example data stream. *5pt*

In synchronous transmission mode, only a maximum end-to-end delay for each unit of a data stream is given. This is useful for real-time sensor data to guarantee that observations are accurate. In isochronous transmission mode, there is also a minimum end-to-end delay specified generally to ensure that a receiver can process incoming data units in real time. It is primarily used for multimedia streaming.

1b Complex data streams consist of multiple substreams (which are not complex) that require synchronization. Give an example of a complex stream consisting of continuous and discrete data streams, and explain how synchronization between those streams can take place. *5pt*

A typical example is a TV data stream consisting of a stream of video frames, two or more audio streams, and a separate discrete stream for subtitling. Synchronization generally takes place by grouping data units into a single packet that is demultiplexed at the receiver. Doing so makes it much easier to ensure that data units of the different streams are played back at the same time.

2a Consider a system that supports remote objects. What information do you need to store in a systemwide object reference to support remote-method invocation? *5pt*

Remote objects are hosted by a server, often by means of an object adapter. What is needed is the transport-level address of that server, along with a local object identifier so that the server knows which (local) object is being referenced. In addition, the object reference will need to refer to client-side code that needs to be present at the client and which implements the object's interface.

2b Explain what needs to be done before a client can invoke a remote object, given it holds an object reference. *5pt*

The client-side software will first need to load the proxy implementation of the object's interface, and instantiate and initialize that proxy. The information to install the proxy is contained in the object reference. For example, the implementation may be available through an archive file named by a URL. When the proxy is installed, the client-side software returns a local reference (i.e., pointer) to the client, and which refers to the interface that the proxy implements. (If you want to be really complete: the server also has to ensure that the object is activated.)

2c Describe a simple and effective way to do automatic garbage collection in the remote-object system just described. *5pt*

A simple solution is to let the object adapter maintain a list of references to clients holding an object reference. When a reference is passed to another client, that client needs to be put on the list as well, for example, by first passing a registration to the object adapter asking for a lease. When the list becomes empty, the object adapter can assume the object is no longer referenced, after which it is destroyed.

3a What is two-phase locking and what does it achieve? *5pt*

2PL refers to the situation that in a transaction locks are never granted after a previous lock for that transaction has been released. It establishes a serialized execution of a collection of concurrent transactions (provided they could be serialized).

3b How can we avoid cascaded aborts in a system of distributed transactions that uses pessimistic timestamp ordering to implement concurrency control? *5pt*

When a scheduler receives an operation $O(T_j, x)$ *on* x *for transaction* T_j *with a higher timestamp than another operation* $O(T_i, x)$ *on* x *for* T_i *that has already been handled by the data manager, it can decide to postpone submitting* $O(T_j, x)$ *until it knows for sure that transaction* T_i *has committed or has been aborted.*

3c Letting each scheduler in a distributed transaction independently implement **strict** two-phase locking may not be very useful. Why not? *10pt*

Strict 2PL is used to avoid cascaded aborts in transactions. Effectively, it establishes that no locks that have been assigned to a transaction T *can be assigned to another transaction before* T *commits or is aborted. In a distributed transaction where the schedulers are distributed across multiple machines, cascaded aborts can be avoided if schedulers release locks at the same time. The whole system falls apart if a single scheduler starts releasing locks when its part of the transaction is finished: there is still a chance that the transaction will be aborted.*

Grading: *The final grade is calculated by accumulating the scores per question (maximum: 45 points), and adding 5 bonus points. The maximum total is therefore 50 points.*