# UNIVERSITY OF DUBLIN

## TRINITY COLLEGE

FACULTY OF ENGINEERING & SYSTEM SCIENCES

DEPARTMENT OF COMPUTER SCIENCE

B.A. (Mod). in Computer Science
Senior Sophister Examination

Trinity Term 2001

4BA8 - Distributed Systems

*Thursday 24th May* *MANSION HOUSE* *14.00-17.00*

Mr. Jim Dowling, Dr. Vinny Cahill, Mr. Brendan Tangney

**Attempt five questions**
(all questions carry equal marks)

Q1. Consider the problem of implementing a fault-tolerant service intended to tolerate the failure of the processors(s) on which it executes using *primary-backup replication.* In particular, you may assume that the processor service has partial amnesia crash failure semantics and that the service manages persistent data whose consistency must be maintained.

i) Outline the design of such a service exhibiting crash failure semantics and required to tolerate up to *k* simultaneous processor failures (i.e., assuming that the only source of failure is processor crash).

(12 marks)

ii) Comment on the extent to which your design of part i) above would be appropriate if the service could exhibit response failures (e.g., assuming that the code for the program is buggy)?

(4 marks)

iii) Comment on the implications of network partition for your design of part i) above.

(4 marks)

Q2. Define the ordering and reliability guarantees provided by each of the following ordered reliable multicast protocols:

- unsafe (i.e., non-dynamically uniform) totally ordered atomic multicast,
- safe (i.e., dynamically uniform) totally ordered atomic multicast,
- unsafe causally ordered atomic multicast,
- causally and totally ordered atomic multicast.

(8 marks)

Given a group communication toolkit that provides ordered member failure notifications and (unsafe) totally ordered atomic multicast, outline the design of a non-blocking version of the two-phase commit protocol. Comment on the extent to which such a commit protocol could be implemented using a multicast protocol providing weaker ordering and reliability guarantees.

(12 marks)

Q3. Outline the design of a new theatre/concert ticket booking system for use in a retail ticket outlet (e.g. HMV, Virgin, etc). The system should be designed to interwork with existing systems operated by individual venues to keep track of ticket sales for that venue. The system should be designed to provide appropriate levels of performance, robustness, security, and scalability.

Your solution should emphasise what you perceive to be the main requirements on the system and the choice of an appropriate distributed systems paradigm (or paradigms) for implementing the system.

(20 marks)

Q4. A campus computer network consists of 2000 publicly accessible PCs permanently connected to the network, in addition there are up to 500 private mobile PCs which are connected to the network from time to time using either fixed wire or wireless connections. There are over 15,000 users of the system.

Assuming that broadcast facilities are available and that on average the processing capacity of the network greatly exceeds the demand placed upon it, design a load balancing system which will distributed user jobs around the system, i.e. when a new task is to be created at a node which is *busy* then the system arranges for it to be created on a suitable remote node. Tasks are not moved once they have begun execution. You may assume the system takes care of all aspects of remote execution such as file handling etc. and that a facility is available to return the load on a node.

Your algorithm should concentrate upon i) efficiently exchanging load information between nodes (8 marks) ii) deciding upon a suitable node on which to create the new task. (5 marks). Justify your design choices.

iii) How would you change your algorithm if broadcast was not supported or if the load placed on the system was close to the overall capacity of the system.

(7 marks)

Q5. i) What is the difference between optimistic and pessimistic distributed algorithms. Illustrate with an example.

(4 marks)

ii) What insights allow the following distributed algorithms to overcome the problems listed.

a) Scale in DNS.
b) Scale in AFS.
c) Packet loss in Frame Relay.
d) Network partition in FDDI.

(4 marks each)

Q6.

i) Remote invocations can fail and the possibility for such failure must be taken into account when writing distributed applications. How are remote failures manifested and what mechanisms are available to programmers in both of the following: Java RMI and OrbixWeb? How are failures in XML-RPC request messages manifested?

(9 marks)

ii) By default, Security Managers in Java RMI Servers don't allow clients to connect to user-level ports (from 1024-65535). Explain the security model in Java (JDK 1.2 and above) and how you would allow a Java RMI Server to grant permissions for clients to connect to a particular port RMI Server's host.

**(6 marks)**

iii) Firewalls are a common problem in building distributed applications. Explain the problems in integrating JavaRMI and CORBA applications with firewalls, and how XML-RPC helps you to build firewall-friendly distributed applications.

**(5 marks)**

Q7. Write brief notes on **ALL** of the following

i) The Fischer-Lynch-Paterson (FLP) impossibility result and its relevance to the designers of dependable distributed systems.

(5 marks)

ii) What is a "nested transaction" and why is support for nested transactions particularly useful in a distributed system?

(5 marks)

iii) In systems design what does it mean to say that "policies and mechanisms should be orthogonal to each other". Illustrate your answer with an example.

(5 marks)

iv) Why is it acceptable for NFS to use delayed write on the client side but not on the server side?

(5 marks)