DUBLIN INSTITUTE OF TECHNOLOGY

DT228 BSc. (Honours) Degree in Computer Science

Year 4

WINTER EXAMINATIONS 2015/2016

Distributed Systems [CMPU4021]

Ms. E. Hatunic-Webster
Dr. Deirdre Lillis
Mr. Kevin Foley

Thursday 12th January 1.00 p.m. - 3.00 p.m.

Attempt **3 questions**
All questions carry **equal** marks
One complimentary mark is available

1. (a) Describe and illustrate with examples *two* design requirements for distributed architectures.

(8 marks)

(b) Using sample code, show how to create a simple TCP server that will receive a Java object of type *Voter*, where the *Voter* class has *name*, *address* and *voterPPS* attributes, all of which are `java.lang.String`.

(12 marks)

(c) With the help of a diagram and examples, explain the Message Oriented Middleware (MOM) paradigm and discuss its advantages and disadvantages.

(13 marks)

2. (a) IP provides a *multicast* facility. Explain what this means, then *discuss* the situations where multicast is used.

(8 marks)

(b) Explain the *three* alternative approaches to external data representation and marshalling.

(12 marks)

(c) The code shown is a partial implementation of a server that broadcasts the time to the clients every few seconds.
Complete the implementation of this code, and write a client that can receive the message (i.e. the time) by passively listening for messages on a `MulticastSocket`. Multiple client processes should be able to run concurrently, with *all* of them receiving the message.

```
import java.net.*;
import java.io.*;
import java.util.Date;

public class  q2C{

   public static void main(String[] args) {
    try {
     DatagramSocket socket = new DatagramSocket(12345);
     DatagramPacket packet;

     InetAddress addrToSendTo = // WRITE THE MISSING CODE

     while(true) {

       byte[] buffer = new Date().toString().getBytes();
       packet = new DatagramPacket(buffer, buffer.length, addrToSendTo, port);
                    socket.send(packet);

        // sleep for a second
        try {
            Thread.sleep((long)(Math.random() * 3000));
          } catch (InterruptedException e) { }

        } catch (Exception e) {}
    }
}
```

(13 marks)

3. (a) The EUelection interface provides two remote methods:

- *vote*: It has two parameters through which the client supplies the name of a candidate (a string) and the 'voter's number' (an integer used to ensure each user votes once only).
- *resultCount:* It has two return parameters through which the server supplies the client with the *name* of a candidate and the *number* of votes for that candidate.

Define the interface to the EUelection service in *Java RMI.*

(8 marks)

(b) The failure model for distributed object systems provides three alternative guarantees for method invocation.

Explain, compare and contrast *maybe*, *at-least-once* and *at-most-once* invocation semantics in this context.

(12 marks)

(c) Discuss in detail *distributed event based* systems, such as those implemented using Jini. In your answer you should explain the roles of the participating objects.

(13 marks)

4. (a) Explain the purpose of UDDI. What are the *four* different data structures that support UDDI? Describe their uses.

(8 marks)

(b) Explain the core technologies of the *web service* approach as middleware and Internet-wide distributed computing and discuss its strengths and weaknesses when compared with competing approaches.

(12 marks)

(c) Managing a *distributed transaction* involves many transaction managers and is made difficult because the state of one system can change suddenly, unknown to the other participants in the transaction.

Using examples and diagrams *show* how the *two phase commit* protocol addresses the problem outlined here.

(13 marks)