

DUBLIN INSTITUTE OF TECHNOLOGY
KEVIN STREET, DUBLIN 8

BSc APPLIED SCIENCES & COMPUTING

YEAR 4

SUMMER EXAMINATIONS 2005

COMPUTER NETWORKS AND DISTRIBUTED SYSTEMS

MR. CIARÁN O'LEARY

DR. BRENDAN O'SHEA

MR. S. MALONE

MAY 2005, 2:30 – 6:00

Answer
Two Questions from Section A
and
Two Questions from Section B

All questions carry equal marks

Section A

1. (a) Java applications and applets run inside a *Java Runtime Environment* provided by the platform or browser.

Clearly explain the role of the Java Runtime Environment, and contrast the execution of Java's bytecode with the execution of natively compiled programs in other languages.

(6 marks)

- (b) Since Java version 1.2 (referred to as Java 2), the responsibility for the security of the Java Runtime Environment is divided between the *verifier*, the *class loader* and the *security manager*.

Explain why security is critical for networked Java applications and in particular for Java mobile code such as Java applets, and describe in detail the role of the *verifier*, the *class loader* and the *security manager* in providing this security.

(9 marks)

- (c) You have been hired by *JavaDating Ltd.*, a company that provides a dating service (meet, fall in love etc.) for Java programmers. The company sells software that allows users connect to other users to whom they can send electronic messages.

Some users of the software are mischievous and send messages to other users while masquerading as someone else. This has resulted in a number of failed relationships and broken hearts – all of which is negatively affecting the company's sales.

Your task is to re-write the software so that it uses Java tools to provide authentication for all messages, so that all message recipients can verify that the sender of the message is who they say they are.

Demonstrate in a step by step fashion how the various different Java tools can be used for an application such as this.

(10 marks)

Continued on next page

2 (a)

```
public class Q2a {
    public static void main(String args[]) {
        (new T()).run();
        (new T()).start();
        (new T()).start();
    }
}
class T extends Thread {
    public void run() {
        for(int i = 0; i < 20; i++) System.out.print(i);
    }
}
```

What output would you expect from the above code. Explain in detail.

(6 marks)

(b) List the different states that a Java thread can be in.

For each state, describe clearly how a thread *enters* and *leaves* the state.

You should refer to the methods of the `Thread` class and the `Object` class in your answer. You must use diagrams and sample code to help illustrate your answer.

(9 marks)

(c) You are required to write the code for the multi-threaded program described below.

- i. The main thread starts off three threads to run concurrently.
- ii. Each thread is responsible for doing the following inside its `run()` method:
 - Adding up all the numbers from 1 to some specific number.
 - Printing out the result of the addition.
- iii. The first thread must add all numbers from 1 to 1000.
The second thread must add all numbers from 1 to 100.
The third thread must add all numbers from 1 to 10.
- iv. You must ensure that while the three threads run concurrently, the *first* thread **MUST** be the *first* to finish and the *second* thread **MUST** be the *second* to finish i.e. the threads must *wait* for each other.

(10 marks)

Continued on next page

3. The following code is needed for all parts of this question.

```
import java.rmi.*;
public class Server {
    public static void main(String[] args) throws Exception {
        Naming.rebind("rmi://localhost:1099/fy", new FactoryImpl());
    }
}
```

```
import java.rmi.*;
public class Client {
    public static void main(String[] args) throws Exception {
        Factory f = (Factory)Naming.lookup("rmi://localhost:1099/fy");
        Adder adder = f.getAdder();
        System.out.println(adder.add(2, 4));
        TodaysDate todaysDate = f.getTodaysDate();
        System.out.println(todaysDate.getDate());
    }
}
```

```
import java.rmi.*;
public interface Factory extends Remote {
    public Adder getAdder() throws RemoteException;
    public TodaysDate getTodaysDate() throws RemoteException;
}
```

```
import java.rmi.*;
import java.rmi.server.*;
public class FactoryImpl extends UnicastRemoteObject
    implements Factory {
    public FactoryImpl() throws RemoteException {}
    public Adder getAdder() throws RemoteException {
        return new AdderImpl();
    }
    public TodaysDate getTodaysDate() throws RemoteException {
        return new TodaysDate((new java.util.Date()).toString());
    }
}
```

```
import java.rmi.*;
public interface Adder extends Remote {
    public int add(int x, int y) throws RemoteException;
}
```

```
import java.rmi.*;
import java.rmi.server.*;
public class AdderImpl extends UnicastRemoteObject implements Adder {
    public AdderImpl() throws RemoteException {}
    public int add(int x, int y) throws RemoteException {return x + y;}
}
```

```
public class TodaysDate implements java.io.Serializable {
    private String date;
    public TodaysDate(String date) { this.date = date; }
    public String getDate() { return date; }
}
```

- (a) Outline in detail all the steps required to compile and run the above application. Explain precisely what happens at each of the steps.
(5 marks)
- (b) Explain in detail the functionality of the above application, and show, using diagrams, which objects and stubs are passed between processes when the application is run. In your explanation you should clearly distinguish between remote and non-remote objects.
(10 marks)
- (c) Describe what is meant by *dynamic class loading*.

Discuss why dynamic class loading of classes from web servers is useful for distributed Java RMI applications.

List and explain in detail each of the steps required to change the above application to one that uses dynamic class loading from a web server.

(10 marks)

Continued on next page

4. (a) Write a Java application that will take a hostname such as `www.comp.dit.ie` as a command line argument. The application should then use the Java API to perform a DNS lookup and print out the IP address of the host.

(5 marks)

- (b) Consider the following code

```
import java.io.*;
import java.net.*;
public class Q4bClient {
    public static void main(String[] args) throws Exception {
        Socket s = new Socket(InetAddress.getLocalHost(), 9999);
        BufferedReader in = new BufferedReader(
            new InputStreamReader(s.getInputStream()));
        PrintWriter out = new PrintWriter(s.getOutputStream(), true);
        out.println("GREETINGS");
        System.out.println(in.readLine());
    }
}

import java.io.*;
import java.net.*;
public class Q4bServer {
    public static void main(String[] args) throws Exception {
        ServerSocket ss = new ServerSocket(9999);
        Socket s = ss.accept();
        BufferedReader in = new BufferedReader(new
            InputStreamReader(s.getInputStream()));
        PrintWriter out = new PrintWriter(s.getOutputStream(), true);
        String message = in.readLine();
        out.println(message);
    }
}
```

This code shows how a client and server process on the same host can exchange a simple text message.

Modify the above code to show how serialised objects belonging to the class `Student` (where a `Student` has a name and an age) can be exchanged instead of the simple text message.

You must also provide the code for the `Student` class.

You must accompany your code with a detailed explanation.

(10 marks)

- (c) Modify the code you wrote in part (b) above so that the server can handle multiple clients concurrently.

You must accompany your code with a detailed explanation.

(10 marks)

Section B

5. (a) Define, using examples, the term *Distributed System*. In your answer you must list the key characteristics of distributed systems. **(5 marks)**
- (b) Discuss the idea of transparency in reference to middleware and distributed systems in general.

You must list five different types of transparency, giving real world examples of where these are evident in distributed systems.

(10 marks)

- (c) You have been hired by the *Dublin Ambulance Service* to design a distributed system for use by the organisation. You are presented with the following requirements specification:

1. All calls from the public are directed to command headquarters in the city centre where they are received by human operators and logged into a database. These calls will include
 - the address where the ambulance is required,
 - the priority level (High, Medium, Low) and
 - the time the call was logged.
2. Ambulance crew have onboard software to interact automatically with the call centre server.
3. It should be impossible for people outside the system to hack in to delete logged calls, modify the details of logged calls or insert bogus calls.

You are required to identify the key requirements for the system just described and select an appropriate architectural model(s). Decisions will also have to be made regarding failure and security models.

Discuss your choices contrasting them with alternatives. Discuss the strengths and limitations of your system.

(10 marks)

6. (a) Using examples, describe the following types of attacks on distributed systems:
- i. Eavesdropping
 - ii. Masquerading
 - iii. Message tampering
 - iv. Replaying
 - v. Denial of service

(5 marks)

(b) Describe in detail, using examples, each of the following approaches to security in a distributed system:

- i. Secret communication with a shared secret key
- ii. Authenticated communication with a server
- iii. Authenticated communication with public keys
- iv. Digital signatures with a secure digest function

For each approach, clearly state the type of scenario when it would be suitable. In your answer you should identify the strengths and limitations of each approach.

(10 marks)

(c) Discuss the importance of security in the modern Internet, in particular on the World-Wide-Web, and provide a detailed description of the approach currently employed by web clients and servers to provide secure communication.

(10 marks)

7. (a) Discuss the importance of *interfaces* in distributed object systems. In your answer you should show how interfaces are provided in Java RMI and also using CORBA's Interface Definition Language (IDL).

(6 marks)

(b) Contrast the *object model* of ordinary local objects and distributed objects under the following headings:

- i. Object references
- ii. Method invocations
- iii. Garbage collection

(9 marks)

(c) Describe the *publish-subscribe paradigm*, and using an example, demonstrate how *event-based systems* are implemented using callbacks in distributed object systems.

(10 marks)

8. (a) Discuss in detail the requirement for *atomicity* in transactions at a single server. You should use examples to justify this requirement.

(8 marks)

(b) Clearly describe what is meant by *nested transactions*, and demonstrate their advantages over *flat transactions*.

(7 marks)

(c) Demonstrate and discuss the operation of *Two-Phase Commit protocol* in distributed transactions.

(10 marks)