Section A — Answer ALL questions

1. State THREE different possible reasons for the use of a distributed system as opposed to using a single computer to do the same task. ALSO, for EACH possible reason give an example of a distributed system that exists for that very reason. (6 marks)

2. a) Define the term **multicasting**. (2 marks)

   b) Explain why support for **multicasting** on either network or middleware level matters to distributed systems developers. (2 marks)

3. a) Define the term **mobile code**. (2 marks)

   b) State ONE advantage and ONE disadvantage of using **mobile code**. (4 marks)

4. a) Explain the term **routing overlay**. (3 marks)

   b) Explain why there is a need to use a **routing overlay** when developing a peer-to-peer system on the Internet. (3 marks)

5. Explain TWO different roles that XML typically plays in Web Services. (4 marks)

6. State what the WS-Addressing standard is for, AND list TWO scenarios where it is used. (6 marks)

7. a) In the context of distributed systems, what do the acronyms MPI and RMI stand for? (2 marks)

   b) State and justify ONE advantage and ONE disadvantage of MPI over RMI. (6 marks)

END OF SECTION A

Section B — Answer TWO questions

8. Consider the following Erlang program:

```
 1  -module(test).
 2  -export([run/0, count/3, stopOne/1]).
 3
 4  run() ->
 5      P1ID = spawn(test, count, ["A",1,7]),
 6      P2ID = spawn(test, count, ["B",100,1]),
 7      receive after 2000 -> ok end,
 8      stopAll([P1ID, P2ID]).
 9
10  stopAll([]) -> done;
11  stopAll([ProcessID | Rest]) ->
12      spawn(test, stopOne, [ProcessID]),
13      stopAll(Rest).
14
15  stopOne(ProcessID) ->
16      ProcessID ! stop.
17
18  count(ProcessName, FirstNumber, Step) ->
19      io:format("~s counts ~p\n", [ProcessName, FirstNumber]),
20      receive
21          stop -> done
22      after 500 ->
23          count(ProcessName, FirstNumber + Step, Step)
24      end.
```

a) State how many Erlang processes are involved in the activities triggered by executing the call `stopAll([P1ID, P2ID])` on line 8. Justify your answer. (4 marks)

b) Write an Erlang function `stopAll2` that differs only in line 12 from the given function `stopAll` in such a way that:

- executing `stopAll2` has almost the same effect as executing `stopAll`,
- `stopAll2` does not use `spawn`. (4 marks)

c) Assuming one replaces line 8 with `stopAll2([P1ID, P2ID])` using the function `stopAll2` described in point (b), state how many Erlang processes are involved in the activities triggered by executing this call. Justify your answer. (3 marks)

d) Explain why there is no benefit in using `spawn` on line 12 in the above program. (5 marks)

(continued...)

(Question 8 continued...)

e) Consider designing a Java RMI program with an analogous structure and functionality to the above Erlang program and answer the following sub-questions related to such a design. Assume that the threads created by commands in the body of function `run` are to be located on a different JVM.

i) Using a collaboration or sequence diagram, show the objects that will be active while the program executes and show the calls made, including their order.

Hints: You will need a factory object. It is expected that your diagram will show around 6 calls.

(6 marks)

ii) Write in full the definitions of all the Java RMI remote interface(s) that will be used in the program.

Hint: The Java code is expected to have around 6–8 lines of code.

(4 marks)

iii) State the TWO most important differences between Erlang messaging and Java RMI messaging.

(4 marks)

9. Consider the following specification of a **RESTful service**. This service allows clients to make enquiries at a specific hotel.
   The server exposes instances of the following types of resource:

   - a collection of user accounts on the URL `"http://my.hotel/users"` supporting method POST
   - a user account on a URL matching `"http://my.hotel/users/{userId}"` supporting GET and PUT methods for existing accounts, represented in XML
   - a collection of enquiries on a URL matching `"http://my.hotel/users/{userId}/enquiries"` supporting method POST
   - an enquiry on URL matching `"http://my.hotel/users/{userId}/enquiries/{enqId}"` supporting method DELETE, represented in XML
   - a response to an enquiry on URL matching `"http://my.hotel/users/{userId}/enquiries/{enqId}/response"` supporting method GET, represented in XML

   a) Assume you are to implement in Java RMI a client-server system where the server offers a functionality analogous to the above RESTful service.

      i) Using a collaboration or sequence diagram, show the objects and method calls in the following interaction between some clients and a server:
      (A) a client locates the server and makes a new enquiry on behalf of some user
      (B) another client supplies a reply to that enquiry.

      Hint: It is expected that your diagram will show around 8–10 objects and 10–12 calls/actions. Clearly group objects that reside on the same JVM. Make at least one use of serialisation.

      (12 marks)

      ii) Write in full the definitions of all the Java RMI remote interface(s) that will be used for accessing the server.

      Hint: The Java code is expected to have around 8–12 lines of code.

      (8 marks)

   b) State and explain the most important differences between HTTP-based RESTful services and Java RMI as a middleware in terms of

      - openness (4 marks)
      - security (2 marks)
      - flexibility (2 marks)
      - scalability (2 marks)

10. a) List and very briefly describe the THREE roles defined by the WS-Notification standard.

notificiation producer

subscription manager

notification consumer = listener

(3 marks)

b) For EACH of the three roles, describe typical operations (if any) offered by those who play this role.

producer: subscribe

consumer: notify

manager: delete, pause, resume

(3 marks)

c) Explain how the WSRF standard is used in the formulation of the WS-Notification standard.

(3 marks)

d) Give a brief overview of the technical steps one has to take to implement a node that plays one of the three WS-Notification roles. Focus on the role of WSDL in this process.

(4 marks)

e) Consider the following distributed system scenario:

> A parcel tracking service is supported by an array of networked computers equipped with scanning devices located in depos and parcel collection points. Customers can register to receive SMS messages whenever their parcel is scanned.

Design a distributed system that delivers the SMS messages to customers. All communication between different nodes in the system must be based on WSDL and WSRF standards. Also, there should be TWO notification arrangements within your design. The two arrangements should work concurrently and one should support the other.

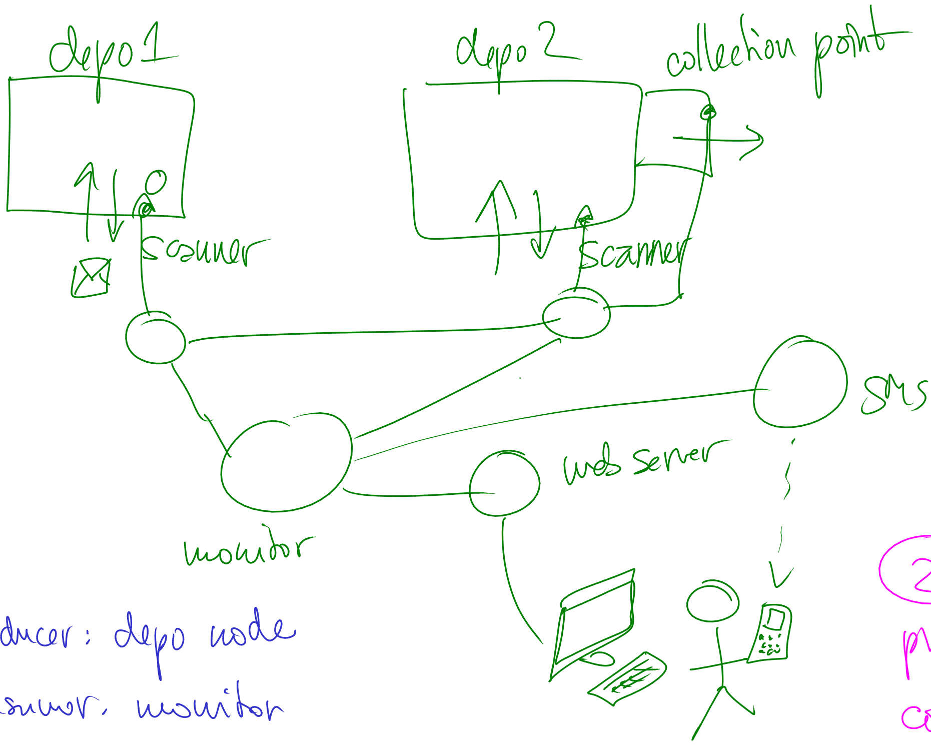
i) Describe all types of nodes in your design and briefly describe their responsibilities.

It is expected that there will be 3–5 types of node. (11 marks)

ii) Identify the two notification arrangements in your design AND indicate for each arrangement and for each notification role, which type of node plays that role.

(6 marks)

END OF EXAMINATION PAPER



1

producer: depo node
consumer: monitor
manager: depo node

2

producer: monitor
consumer: SMS sender
and Web Server
manager: monitor