FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Departamento de Informática

Licenciatura / Mestrado em Engenharia Informática
Sistemas Distribuídos – 1º teste, 22 de Outubro de 2012
1º Semestre, 2012/2013

NOTAS: Read carefully eaxh question before answering. **The test is closed book. The duration of the test is 1h30min**. The test contains **4** pages.

NAME: _____ **NUMBER**: _____

1) Consider the context of practical work, where a set of servers provides access to files over RMI and Web Services (SOAP) interfaces. Complete the code in annex A, an excerpt of the code of the system, with the adequate code given the comments in the code.
2) Assign **[T]rue** or **[F]alse** to each sentence (**note: incorrect questions will deduct**):

___ Middleware systems help addressing heterogeneity problem.

___ In a distributed system it is impossible to a machine to know if other machine has failed only through sending/receiving .

___ TCP implements a model of volatile communication (in the presence of failures).

___ UDP implements a model of communication with delivery of messages by the same order of emission.

___ A *message queuing system* implements a model of synchronous communication.

___ Client/server model does not present good fault tolerance because it has a unique failure point.

___ In client/replicated server model, if all operations modify server state, the throughput of the system cannot improve with adding more replicas.

___ A non-structured *peer-to-peer* model allows efficient search of resources.

___ A structured *peer-to-peer* model (DHT) allows efficient search of resources given its name.

___ *BitTorrent* combines client/server and peer-to-peer models.

___ In *BitTorrent*, the fact that a server sends some packets without requiring to receive packets from the receiver, allows a new client to join file sharing.

___ The main advantage of a proxy close to the server is to improve latency of reply to clients.

___ A system that fails often can have high reliability.

___ Authentication of principals (elements of the system: client and server) is fundamental to execute access control in a distributed system,

In the context of remote method invocation:

___ Marshaling (coding) an object in Corba typically uses less space than marshaling the same object in XML.

___ Sending messages in an intermediate format is typically more efficient in terms of required computation when compared with sending messages in the sender format.

___ To implement "exactly once" semantics it is necessary to store information in stable memory.

___ Java RMI uses "at most once" invocation semantics.

___ A remote reference to a server stored in the *rmiregistry* is automatically collected when the server stops executing.

___ *Rmiregistry* executing in machine X allows to register server executing in machine Y.

___ In Java RMI, invocation from different clients execute sequentially – an invocation only starts after the previous one completes.

___ In Java RMI, a server object is *garbage collected* if there is no reference to it (local or remote).

___ WSDL includes information about exceptions that can be thrown by a method.

___ Unlike RMI, the invocation of a remote method in Web Services never throws exceptions dur to communication problems.

3) Suppose that you intend to implement a system similar to Twitter, with a large number of users that maintains the list of messages that the user has added.

   a) What architecture would you use to implement such system. Justify.

Client/server / Client/replicated server / Client/partitioned server

   b) Would it be interesting to additionally maintain users' message in an edge server present in servers at ISPs (e.g. with Akamai)? Justify with the base in a realistic usage scenario.

Usage scenario:

Answer: Yes/ No , because

4) Using idempotent operation simplifies the implementation of a remote invocation protocol that intends to guarantee that an operation has been executed in the server? Justify.

Yes, because… / No, because…

ANNEX A

```
/** Exception meaning that a file/directory does no exist */
public class NotFoundException extends Exception { ... }

/** Interface implemented by RMI server */
public interface IFileServer extends                    {
    /** Lists the names of files in a given directory. On error, throws NotFoundException     */
    public String[] dir( String path) throws NotFoundException,                    ;
    …
}

/** Class for WS server */

public class FileServerWS {
    /** Lists the name of files in a given directory. On error, throws NotFoundException  */

    public String[] dir( String path) throws NotFoundException { … }
    …
}
/** Main class for client */
class FileClient {
    /** Returns RMI remote reference for a RMI server given its URL. On error, returns null */
    public IFileServer getRMIRef ( String url) {
```

```
    }
    /** Returns remote reference for WS server given its URL. On error, returns null.
    NOTA: ws.FileServerWS is the type created by wsimport for server FileServerWS */
    public ws.FileServerWS getServerRef ( String url) { … }
    /** Returns URL for servers – RMI: rmi://server/name ; WS: http://server/name*/
    public String[] servers() { … }
    /** Returns a list of files/directories in path in all servers.
     * Returns null on error accessing all servers.
     * IMPORTANT NOTE: all element must have the format: nome@url */
    protected List<String> dir( String server, String path) {
        List<String> res = new ArrayList<String>();
```

```
} }
```