RMI exercise

Daniel Hagimont

Daniel.Hagimont@irit.fr

USTH

The objective of this lab is to practice with Java RMI to implement a simple application where a graphical applet allows to enter the coordinates of a person (name, email) and to register them in a remote server. The GUI of the applet allows selecting the server in which the user wants to register the person. We will manage only 2 servers.
The application also allows to lookup for a person given his name, indicating the server where this lookup should be performed.

If the person is not found in the indicated server, the search should be propagated to the other server.
The interface of the server class is the following:

```
public interface Pad extends Remote {
        public void add(SRecord sr) throws RemoteException;
        public RRecord consult(String n, boolean forward) throws RemoteException;
}
public interface SRecord extends Serializable {
        public String getName ();
        public String getEmail ();
}
public interface RRecord extends Remote {
        public String getName () throws RemoteException;
        public String getEmail () throws RemoteException;
}
```

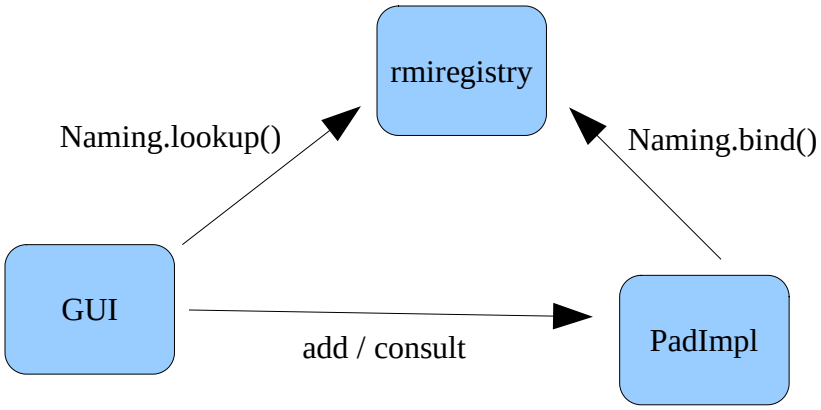
The registering of a person (add() method) relies on serialization to send to the server a copy of an object which implements the SRecord interface.
A lookup for a person (consult() method) returns a remote reference to a RMI object which implements the RRecord interface ; the applet can then use this remote reference to access the email address of the person with a remote invocation on that reference.
The consult() method includes a boolean parameter (*forward*) which indicates whether the request should be propagated to the other server if the person is not found (in order to prevent looping if the person does not exist).

You are given:
- Pad.java : the interface of the server
- SRecord.java : the interface of the object which is serialized and passed to the add() method
- RRecord.java : the interface of the RMI object returned by the consult() method
- GUI.java, java.policy, page.html : the applet which implements the GUI of the application

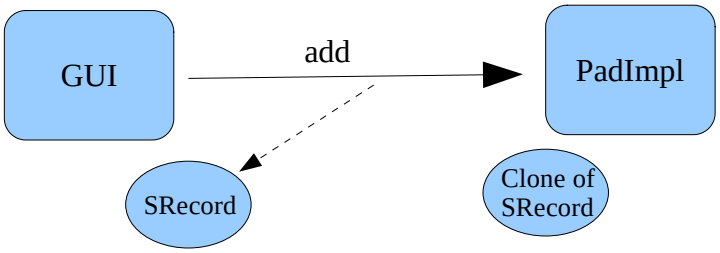
To run the applet:
- in standalone mode: java GUI
- with the appletviewer : appletviewer -J-Djava.security.policy=java.policy page.html

You have to:
- implement PadImpl.java : the server class
- implement SRecordImpl.java : the class of the serialized object
- implanter RRecordImpl.java : the class of the RMI object returned by the server
- modify GUI.java : to implement calls to the servers

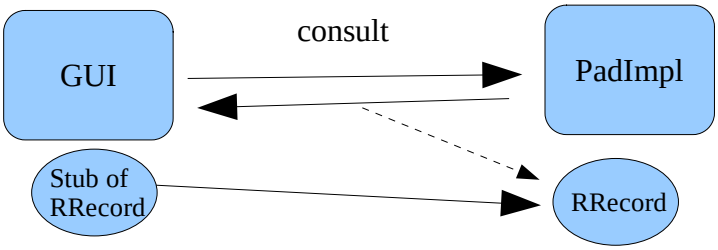
==========



When instanciated, the PadImpl registers (bind()) a stub in the rmiregistry. The GUI lookup the stub from the rmiregistry. It can then invoke methods (add/consult) on the PadImpl.



When add() is invoked, it takes as parameter a SRecord object which is serializable, so a copy of the object is passed.



When consult() is invoked, a RRecord object is returned, and since it is a remote object, a stub is returned, which makes the object accessible remotely from the GUI. The GUI can then invoke getEmail() remotely on the RRecord object.