Distributed Programming II

A.Y. 2015/16

Sample Final Test

All the material needed for this test is included in the *.zip* archive where you have found this file. Extract the archive to an empty directory where you will work, and copy your solution of Assignment 4 into this directory.

The test consists of a programming exercise (6 points) and a question (4 points). The exam can be passed only if the programming exercise solution passes the mandatory tests (at least test `testRightActionTaken`). In this case, the proposed mark will be the sum of the points got for the test and a base evaluation mark in the range 16-20, assigned on the basis of the evaluation of the submitted assignments (16 points granted because your assignments passed the mandatory tests at submission time and 4 extra points based on the evaluation of one extra aspect of your assignments).

Programming exercise

1. Modify your server so that it also includes the implementation of take over operations. The URL used by the server for receiving take over requests can be selected at your choice.

   The service must perform take over operations in such a way that the following requirements are satisfied:

   - Taking over an action is possible only for actions that have not yet been taken over.
   - The first client that tries to take over a given action succeeds while the next clients trying to take over the same action fail.

   The server must be able to execute operations concurrently and allow at least up to 5 requests to be executed concurrently.

2. Create a client for the service that offers take over operations. The client must take the form of a Java library that implements the interface *it.polito.dp2.WF.lab4.WFTakeOverClient*, given in source form. The interface is self-explaining (look at the comments). The client class must be named *it.polito.dp2.WF.sol4.client3.WFTakeOverClientImpl*, and must have a default constructor (i.e. a constructor without arguments).

Apart from these new specifications, all the specifications given for Assignment 4 still apply.

In particular, all classes must be developed in the same packages used for the solution of Assignment 4 and stored in the same directories as the solution of Assignment 4. Make sure that the *ant* script `[root]/sol_build.xml` still works for the compilation of your new solution and modify it as necessary. You can assume that, when your client is compiled and run, your web services have already been deployed (i.e. your `WorkflowServer` application has already been started).

Question

Write the lines of code of your serializer of Assignment 2 where you perform the marshal operation. Then explain the meaning of each line.

Explain how in the code of your server you avoid that two clients that are performing the take over operation concurrently on the same action succeed both.

The answer to this question must be written in a text file named `[root]/answer.txt`

Correctness verification

In order to pass the exam your solution must pass at least the mandatory tests that are included in the archive (the sources are available in the folder `it.polito.dp2.WF.lab4.tests`). These tests can be run by the ant script `buildTest0.xml` included in the *.zip* file, which also compiles your solution and runs the client and the server by calling your ant scripts. The command for running the tests is

```
ant –f buildTest0.xml run-final-test –Dseed=XXXX
```

There is a total of 4 junit tests. In order to pass the exam it is necessary to pass at least the `testRightActionTaken` test.

Submission format

A single *.zip* file must be submitted, including all the files that are part of your solution (including the files of your solution of Assignment 4 that you are re-using). The *.zip* file to be submitted must be produced by issuing the following command (from the `[root]` directory):

```
ant –f buildTest0.xml make-final-zip
```

Important: check the contents of the zip file named `solution.zip` after having run the command!