Distributed Programming II

A.Y. 2016/17

Final Test 3

All the material needed for this test is included in the folder where you have found this file. In order to import correctly the Java project of the exam into Eclipse, follow these steps: go to `File -> Open Projects From File System`; click the `Directory` button next to `Import source`; find the project's folder into the Browse window. When the project has been imported, copy your solution of Assignment 3 into this folder.

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 one of the tests specific for this assignment). 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 a maximum of 4 extra points based on the evaluation of one extra aspect of your assignments).

Programming exercise

1. *Modify* your service developed in Assignment 3 so that it is possible for special administration clients to put the service into a *maintenance* state and to reset it again into the *normal* state. When in maintenance state, the service must not accept new operation requests from normal clients (neither for getting information, nor for creating resources), but requests that are already in progress must be completed. Of course, requests for changing the maintenance state, coming from administration clients, must be accepted. Administration clients are supposed to change the maintenance state but not to perform other operations. Optional: (i) if a new request arrives from a normal client and the service is in maintenance state, the service must inform the client that the operation is impossible because the service is in maintenance state (i.e. the client should be able to distinguish this error condition from other possible errors).

   No client authentication must be implemented and all the other features of the service must remain unchanged.

2. *Create a new* client for your service, named client3, which implements the interface `it.polito.dp2.NFFG.lab3.test3.MaintenanceClient`, given in source form (the interface is self-explaining, look at the comments).

   Create a factory class for your client3 named `it.polito.dp2.NFFG.sol3.test3.client3.NFFGClient3Factory` that extends the abstract factory `it.polito.dp2.NFFG.lab3.test3.NFFGClient3Factory` and, through the method `newNFFGClient3()`, creates an instance of your concrete class that implements the `it.polito.dp2.NFFG.lab3.test3.MaintenanceClient` interface. All the classes of your client3 must be in the package `it.polito.dp2.NFFG.sol3.test3.client3`. The actual base URL used by the client class to contact the service has to be read as the value of the `it.polito.dp2.NFFG.lab3.URL` system property.

3. Optional: modify your client2, and its Factory if necessary, so that in case the service is in the maintenance state, the method `newNFFGVerifier()` throws the exception `it.polito.dp2.NFFG.lab3.test3.MaintenanceException` (which is a specialization of `it.polito.dp2.NFFG.lab3.NFFGVerifierException`).

Apart from these new specifications, all the specifications given for Assignment 3 still apply.

Modify the *ant* script `[root]/sol_build.xml` so that it includes the compilation of your new client. As usual, you can assume that, when the tests are executed and your client is compiled and run, your web service has already been deployed. Note that if you run the build-client target yourself (rather than letting the run-tests do it), this assumption may be false and you must make sure the service has been deployed.

Question

Explain how you avoid race conditions in the operation(s) that change the maintenance state. In your explanation, refer to the code of your server and copy the most significant lines of code of your server that show what you are explaining.

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

Correctness verification

In order to pass the exam you have to implement at least points 1. and 2., and your solution must pass at least the mandatory tests that are included in the archive (the original tests of Assignment 3 plus at least one test from the additional junit test suite `it.polito.dp2.NFFG.lab3.test3.tests.NFFGTests3`). These tests can be run by the ant script `buildTest3.xml` included in the exam folder, which also compiles your solution, packages and deploys your service to Tomcat, and runs your clients.

The Tomcat server and Neo4J must be both running when the tests are launched. In particular, start Tomcat from the graphical interface of Xampp, in order to have the proper user configuration (i.e., user `root` – password `root` - with role "`manager-gui, manager-script`" at least). The Neo4J database, instead, must be started from command line. A distribution of the data-base is already available and properly configured under the `Z:\DP2_23_06_2017\neo4j-community-3.1.1` folder. In order to start Neo4J, run the following command from `Z:\DP2_23_06_2017\neo4j-community-3.1.1\bin`:

```
neo4j.bat console
```

The command for running only the tests specific for this assignment (with the exclusion of the original ones of Assignment 3) is

```
ant –f buildTest3.xml run-tests –Dseed=XXXX
```

The total number of junit tests in this case is 2.

The full set of tests (including the original ones of Assignment 3) can be launched by running the `run-tests-full` target:

```
ant –f buildTest3.xml run-tests-full –Dseed=XXXX
```

As this target may take a long time to complete, initially you are suggested to use the `run-tests` target.

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 3 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 buildTest3.xml make-final-zip
```

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