Distributed Programming II

A.Y. 2017/18

Final Test 4

All the material needed for this test is included in the folder where you have found this file. Copy your solution of Assignment 3 into this folder.

The test consists of a programming exercise (total 6 points) and two questions (total 4 points). The exam can be passed only if the programming exercise solution passes the mandatory tests (see Correctness Verification). 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 the design and implementation of your service developed in Assignment 3 so that it also provides the possibility for clients to specify, for the deployment of a new NF-FG or for the addition of a node to an NF-FG, that the operation should be done without allocating nodes on one or more hosts specified by the client. For example, a client may request that a NF-FG be deployed without allocating nodes on Host0. Of course, the service must take this request into consideration.

   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.NFV.lab3.test4.NfvClient3`, given in source form (the interface is self-explaining, look at the comments). Initially, the client must assume that the set of unwanted hosts is empty. Then, the set can be modified through the interface. All the classes of your client3 must be in the package `it.polito.dp2.NFV.sol3.test4.client3`, but this new client should re-use classes from the other previously developed clients. In the same package, create a factory class for your client3 named `NfvClient3Factory` that extends the abstract factory `it.polito.dp2.NFV.lab3.test4.NfvClient3Factory` and, through the method `newNfvClient3()`, creates an instance of your concrete class that implements the `it.polito.dp2.NFV.lab3.test4.NfvClient3` interface. The actual base URL of the web service to be used by client3 is by default `http://localhost:8080/NfvDeployer/rest/` but it can be configured through the client interface.

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 works with the modified service and it includes the compilation of your new client. As usual, you can assume that, when your client is compiled and run, your web service has already been deployed.

Questions

Describe and explain how your service implementation validates data coming from clients, so that invalid data are refused.

Explain if and how, in the design of your service, you limited the number of interactions and you avoided that clients are obliged to receive huge response messages.

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

Correctness verification

In order to pass the exam you have to complete 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 the test `testNodeAddition` from the additional junit test suite `it.polito.dp2.NFV.lab3.test4.tests.NFVTests4`). These tests can be run by the ant script `buildTest4.xml` included in the *.zip* file, 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. The command for running only the tests specific for this assignment (with the exclusion of the original ones of Assignment 3) is

```
ant -f buildTest4.xml run-tests -Dseed=XXXX
```

The total number of junit tests in this case is 3.

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

```
ant -f buildTest4.xml run-tests-full -Dseed=XXXX
```

As this target may take a longer 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 buildTest4.xml make-final-zip
```

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