MTAT.03.229 – Enterprise System Integration

Regular Exam – 19 January 2015

(50 points)

Notes:

- The exam is open-book and open-laptop. Web browsing is allowed
- You are not allowed to communicate with anyone during the exam in any way (except with the lecturer).
- You may submit your exam on paper, or electronically using the "Submit" button in the course Web page.The solution to Part 1 (zip file containing the eclipse project) must be submitted using the "Submit" button in the course Web page. The solution to Part 2 can be submitted on paper or electronically. When electronically, you must include all files (e.g. PDF, DOC or DOCX, MD) within the eclipse project used in Part 1.
- If you find that there is not enough information in the text below to answer a given question, and you need to make additional assumptions, please write down your assumptions together with your answer.

Part 1. BuildIt: Testing & Code Coverage

One of the programmers behind RentIt's information system has suddenly left the company. Although the software seems properly implemented (and nicely running), there are no tests for some critical parts of the system. Therefore, you are requested to take over and implement part of the tests.

You can clone the git repository with the current implementation from https://bitbucket.org/lgbanuelos/esi2014-exam2 (To ease your task, the repository does not contain the full implementation but the part of the program that you are concerned with). The code is distributed in the form of two eclipse projects, namely testingexample2-jdk7 and testingexample2-jd8. You should choose either one depending on the version of Java that you have installed in your computer. The eclipse project contains already a skeleton for your testing code (i.e. class "PurchaseOrderManagerTests") and a dataset, i.e. file "dataset.xml", which you can use in your tests.

Task 1 [15 points] Concretely, you are requested to implement the tests to ensure a 100% of code coverage on class "PurchaseOrderManager".

HINTS:

1. Given the number of conditional paths, you should implement at least 4 tests.
2. The project's "pom.xml" is configured in the same way that we used for continuous integration. Therefore, it is possible to run "cobertura" to determine the current code coverage percentage. If you have maven installed in your computer, you can simply run "mvn cobertura:cobertura" within the project directory. Alternatively, you can run the same maven target from STS (if you don't know how, you can ask the lecturer).

Part 2. RESTful API for CAD Simulations

Engineers at automobile design company CaaS routinely run simulations of new automobile designs or automobile component designs in order to study their mechanical properties. Engineers design the automobile or automobile components using a Computer-Aided Design (CAD) tool. The design is exported as a CAD file and uploaded to the main server of the company's computer cluster via an FTP client. Engineers then login to this main server and schedule their simulation on the computer cluster using a command-line tool (the command-line tool is a Python script). Simulations may take between one and 12 hours depending on the complexity of the design (average of 6 hours). However, since the capacity of the computing cluster is limited, it takes several hours between the moment a simulation is scheduled by an engineer and the moment it starts. To monitor the status of a simulation, engineers need to login to the server and use another command-line tool. Engineers can also cancel a simulation using yet another command-line tool. Finally, engineers get an e-mail when their simulation is completed and they need to login to the main server to retrieve the file containing the simulation outputs. They then load this file into their CAD tool in order to analyze the results. Around 20 simulations per day are executed on CaaS's computer cluster. CaaS has 20 engineers who routinely run simulations and another 20 engineers who run simulations occasionally.

Several problems have been identified with the current way of running simulations. First, the process is cumbersome for engineers, and it is difficult to teach it to new engineers who join the company. As CaaS is foreseeing a significant expansion in the next year, and plans to double its number of engineers, it will become crucial to make this process simpler. Also, given this planned expansion of the company, it is clear that the computer cluster is reaching its limits and scaling up the cluster would be expensive. To scale up, CaaS is planning to rent servers from cloud computing providers on a per-hour basis, to run some of the non-critical simulations (simulations of the most critical vehicle models would still be done in the local computer cluster). The fact that external cloud-based servers will be used will make the procedure for starting and monitoring simulations more complex and engineers are against this additional complexity, given that it takes away time from them that could be used for more productive purposes. Finally, a third issue with the current simulation process is that no history of past simulations is kept. Sometimes one engineer runs a simulation, fetches the results into his/her laptop, and when another engineer wants to inspect the results of this simulation, it is difficult for them to retrieve these results. It has become evident that engineers need a searchable and browsable archive of all previous simulations.

In parallel, IT managers intend to start internally billing (also known as "IT chargeback") on a monthly basis for completed simulations, in order to offset growing computing costs. The cost of a completed simulation will be proportional to its duration, and will depend on whether the simulation is "critical" or "non-critical".

To address the above needs, you are asked to design a RESTful Application Programming Interface (API) for a simulation environment to be deployed in the main server of the computer cluster. Engineers will be able to start, monitor and retrieve the output of their simulations directly from their CAD tool by means of a special CAD tool plugin that will be developed for them. This plugin will be developed in Java Standard Edition (Java SE), which is the plugin development technology supported by the CAD tool. The plugin will interact with the server-side simulation environment via HTTP operations, using JSON as the resource representation format.

Given the inherent complexity of a CAD file, its content is not directly as part of the JSON file to be used in the RESTful interactions. Instead, the CAD files are stored in a “document management system” (DMS). The DMS associates a URL to each CAD file, which can then be used for retrieving a given CAD file by using a HTTP GET request on the corresponding URL. Simulation results will also be stored in the DMS – the DMS provides operations for pushing files into it.

You are requested to design an API for the simulation environment. The interface should provide all operations that the CAD tool plugin would need to invoke to create and manage simulations on the server-side simulation environment. This includes the ability to create a simulation; to cancel a scheduled or a running simulation; to retrieve the current state of simulations (e.g. scheduled, running, completed, …); to view a simulation (including the link to the simulation output if completed); and to query the full history of created simulations. When retrieving the history of simulations, one should be able to filter the simulations by one or multiple of the following criteria:

- The time window when the simulations were created (captured by a start date and an end date of the time window)
- A keyword, which may be the simulation’s unique identifier, part of the simulation’s name or part of the simulation’s description given by the engineer who started the simulation.
- The status (e.g. only “completed” simulations or only “scheduled”)
- The creator of the simulation (username).
- The criticality of the simulation (see above).

Task 2 [5 points]. Design a domain model for the simulation environment.

Task 3 [5 points]. Based on the domain model, design a *resource model* using the RESTful service design method presented in the lectures.

Task 4 [5 points]. Specify a state machine capturing the lifecycle of a simulation. Each transition triggered by an external operation should be labelled with the name of the operation, the corresponding HTTP verb and URL of the resource.

Task 5 [15 points] Specify a RESTful interface for the simulation environments by means of an apiary blueprint. For each operation on a resource, you should specify an example of a request/response, using “links” where appropriate.

Task 6 [5 points]

a. Engineers wish to see the queue of simulation in order to estimate when their simulation will start. Is the API you designed able to retrieve this queue and if so what query should be used to do so? If not, explain what should be added to the API to allow engineers to retrieve the simulation queue?

b. What query(ies) of the proposed API could be used to determine how much a given team of engineers have spent in a given month (e.g. in December 2014) running simulations, in other words how many simulation-minutes should be cross-charged to the team in question?