Home Exam – Software Engineering – D7032E – 2015

Teacher: Josef Hallberg, josef.hallberg@ltu.se, A2588

Instructions

- The home exam is an individual examination.
- The home exam is to be handed in by **Friday October 30, at 17.00**, please upload your answers in Fronter (in the Submissions/Home Exam folder) as a compressed file (preferably one of following: rar, tar, zip), containing a pdf file with answers to the written questions and any diagrams/pictures you may wish to include, and your code. Please name the file with your name and group name/number, it makes it easier for me to find you in the list. If for some reason you can not use Fronter (should only be one or two people) you can email the Home Exam to me. Note that you can NOT email a zip file since the mail filters remove these, so use another compression format. Use subject "**D7032E: Home Exam**" so your hand-in won't get lost (I will send a reply by email within a few days if I've received it).
- Every page should have your full **name** (such that a singular page can be matched to you) and each page should be numbered.
- It should contain ***original*** work. You are NOT allowed to copy information from the Internet and other sources directly. It also means that it is not allowed to cheat, in any interpretation of the word. You are allowed to discuss questions with class-mates but you must provide your own answers.
- Your external references for your work should be referenced in the hand in text. All external references should be complete and included in a separate section at the end of the hand in.
- The language can be Swedish or English.
- The text should be language wise correct and the examiner reserves the right to refuse to correct a hand-in that does not use a correct/readable language. Remember to spellcheck your document before you submit it.
- Write in running text (i.e. not just bullets) – but be short, concrete and to the point!
- Use a 12 point text size in a readable font.
- It's fine to draw pictures by hand and scanning them, or take a photo of a drawing and include the picture; however make sure that the quality is good enough for the picture to be clear.
- Judgment will be based on the following questions:
  - Is the answer complete? (Does it actually answer the question?)
  - Is the answer technically correct? (Is the answer feasible?)
  - Are selections and decisions motivated? (Is the answer based on facts?)
  - Are references correctly included where needed and correctly? (Not just loose facts?)

Total points: 25	
Grade	Required points
5	22p
4	18p
3	13p
U (Fail)	0-12p

Good luck! /Josef

Questions

1. The Software Engineering context of Software Architecture (3p, max 0.5 pages)

Describe the steps in a best practices example of designing an architecture for a large software from scratch. There may be different best practices for different types of software/teams/software development models, so write a sentence describing the type of software/team/model you are using in your example. Which stakeholders/roles are involved? What iterations are made? Which artifacts (documentation etc.) are generated? Note: you only need to describe as far as to delivering the first version of an architecture specification.

2. System Architecture (4p, max 0.5 pages excluding diagram)

For the following problem, describe the system architecture in the following form:

- Name one *architectural pattern* (hint: see lecture 12) that you will use (not design pattern).
- Draw a diagram that describes your system architecture.
- Quickly explain in words how the system works.
- State the three most important advantages of using this architecture.
- State the two most important disadvantages of using this architecture.

Problem: In a photo-album application, there are many filters: sort by theme, size, time, friends in the photo, etc. Several filters can be applied at once to help the user gain a quick overview of the photo album. The application will be developed by three different teams with different expertise (one team, for example, are expert at image recognition and will develop the face-detection feature). It should be possible to easily share pictures with friends using the most common social media platforms.

3. Architecture design / Software maintenance (2p, max 0.5 pages)

Discuss why we want low coupling and high cohesion when designing a software architecture. Motivate your answer, and give two examples of quality attributes in which low coupling and high cohesion are of extra importance.

4. Testing (2p, max 0.5 pages)

Why is it that many companies write more code for the tests than they do for the application/software itself? How can such behavior be supported and motivated from an economical standpoint?

5. Implementation (1p, max 0.5 pages)

Discuss when *"just coding"* (as opposed to designing the architecture before coding) may be appropriate and when it is not. Motivate your answer clearly.

6. Quality Attributes, Design Patterns, Documentation, Re-engineering, Testing (13p)

A newbie programmer has created a simple dungeon-crawler game called Dragon Treasure. The newbie is quite happy with the look and feel of the game, but wants it to be more flexible. There are a few features the newbie wishes to add, and a few bugs that newbie wants to have fixed. Please help the newbie by re-engineering his code and help him create a better design and code, which is easier to understand. The documentation consists of a drawing of the dungeon. The source code is provided in the `Application.java` file (compile with: javac Application.java run with: java Application).

The newbie wants to be able to add a KeyListener in the future so that he doesn't need to press [Enter] after choosing a command. However, this is currently not possible since the game runs in the console. The newbie thinks that in the future he might want add a graphical user interface, which would then make it possible to also use a KeyListener. The newbie wants you to make it easy to add a KeyListener (or similar mechanism depending on your chosen programming language) in the future, and doesn't want to make changes in lots of places when adding it.

The newbie wants to fix the following bugs: Potions keep re-spawning after they have been used, and after the Dungeon hero returns back to the room in which the potion was originally located. This is not supposed to happen. Furthermore, the beast and the dragon keep re-spawning with 0 health after they have been defeated if the dungeon hero returns to that room. The dungeon hero seems to also be able to steal the treasure multiple times. Please help the newbie to fix these bugs.

The newbie wishes the software architecture design was better made so that it would be easy to add new rooms, new monsters, new treasure, etc. in the future. Your task is to completely re-engineer the code and pay close attention to good software architecture design and coding best practices (keeping in mind the quality attributes: coupling, cohesion, sufficiency, completeness, primitiveness - http://atomicobject.com/pages/OO+Quality). You should design your architecture for *Modifiability* (see slides for lecture 5) and *Comprehensability* (make the code easy to read and understand), and utilize code reuse and design-patterns where appropriate. You are of course allowed to add additional classes and change as much of the code as you wish/need, as long as the game plays and looks roughly the same. Follow the re-engineering principles presented lecture 9.

For each of the rooms in the game (alternatively, for the full play through) you should also add unit-tests, which verifies that the game runs correctly (it should result in a pass or fail output, preferably color coded). The syntax for running the unit-test should also be clearly documented. Note that the implementation of the unit-tests should not interfere with the rest of the design. Hint: it's possible to generate KeyEvents.

Examination criteria - Your code will be judge on the following criteria:

- Whether it is easy for another developer to add support for new rooms, monsters, treasure, etc.
- How easy it is for another developer to understand your code by giving your files/code a quick glance.
- To what extent the coding best-practices have been utilized.
- Whether you have paid attention to the quality metrics when you re-engineered the code.
- Whether you have used appropriate design-patterns in your design.
- Whether you have used appropriate variable/function names.
- To what extent you have managed to clean up the messy code.
- Whether program uses appropriate error handling and error reporting.
- Whether the code is appropriately documented.
- Whether you have created the appropriate unit-tests.

If you are unfamiliar with Java you may re-engineer the code in another object-oriented programming language. However, instructions need to be provided on how to compile and run the code on either a Windows or MacOS machine (including where to find the appropriate compiler if not provided by the OS by default).