Software Engineering Methods 2016

Instructor: Dr. Alberto Bacchelli — Assignment 1

Teaching Assistants: Liam Clark, Robin van Heukelum, Bart de Jonge, Jean de Leeuw, Thomas Overklift, Mark van de Ruit, Wouter Smit, Martijn Steenbergen, Gijs Weterings — Week 03

In this assignment you apply (1) **Responsibility Driven Design** and (2) **Software Modeling with UML**. To correctly complete this assignment you **must**:

- Carry out the assignment with your team only–unless otherwise stated. You are free to discuss solutions with other teams, but each team should come up its own personal solution. A strict plagiarism policy is going to be applied on all the artifacts submitted for evaluation.
- Complete the assignment by following the SCRUM methodology. Regarding this, you must push the following documents on the main branch of your GitHub repository by the specified deadlines and tag the commit that your TA has to consider for grading:
  - A sprint backlog for this assignment, using the template "Sprint backlog - template" available on Blackboard, by **Sep 19, 2016 @ 10:45AM**.
  - A sprint retrospective for this assignment, using the template "Sprint retrospective - template" available on Blackboard, by **Sep 23, 2016 @ 17:55PM**.
- Provide solutions to the exercises. Each solution will consist of changes to the source code of your project and/or explanations (*e.g.*, of decisions taken):
  - The explanations must be written in a PDF file with the name: *Group[id on google spreadsheet]-[AssignmentNumber].pdf*[1]
  - Changes and explanations must be pushed to the master branch of your GitHub repository by **Sep 23, 2016 @ 17:55**[2] and the commit that you TA has to consider must be tagged.

Exercise 1 - The Core (18 pts)

1. Following the Responsibility Driven Design, start from your requirements (without considering your implementation) and derive classes, responsibilities, and collaborations (use CRC cards). Describe each step you make. Compare the result with your actual implementation and discuss any difference (*e.g.*, additional and missing classes). (**4 pts**).
2. Following the Responsibility Driven Design, describe the *main* classes you implemented in your project in terms of responsibilities and collaborations (**4 pts**).
3. Why do you consider the other classes as less important? Following the Responsibility Driven Design, reflect if some of those non-main classes have similar/little responsibility and could be changed, merged, or removed. If so, perform the code changes; if not, explain why you need them (**4 pts**).
4. Draw the class diagram of the aforementioned main elements of your game (do not forget to use elements such as parametrized classes or association constrains, if necessary) (**2 pts**).
5. Draw the sequence diagram to describe how the main elements of your game interact (consider asynchrony and constraints, if necessary) (**4 pts**).

[1] *e.g.*, a correct name would be: *Group0-1.pdf*.

[2] Solutions sent within the first 24 hours after the deadline will be given 50% of the points they would normally get. Solutions sent after 24 hours from the deadline will not be graded.

Exercise 2 - UML in practice (9 pts)

1. What is the difference between aggregation and composition? Where are composition and aggregation used in your project? Describe the classes and explain how these associations work (**3 pts**).

2. Is there any parametrized class in your source code? If so, describe which classes, why they are parametrized, and the benefits of the parametrization. If not, describe when and why you should use parametrized classes in your UML diagrams (**3 pts**).

3. Draw the class diagrams for *all* the hierarchies in your source code. Explain why you created these hierarchies and classify their type (*e.g.*, "Is-a" and "Polymorphism"). Considering the lectures, are there hierarchies that should be removed? Explain and implement any necessary change (**3 pts**).

Exercise 3 - Simple logging (33 pts)

1. Extend your implementation of the game to support logging. The game has to log all the actions happened during the game (*e.g.*, player moved Tetris piece from position X to position Y). The logging has to be implemented from scratch without using any existing logging library. Define your requirements and get them approved by your teaching assistant. (**20 pts**).

2. During the analysis and design phases of this extension use responsibility driven design and UML (push to the repository a *single* PDF file including all the documents produced) (**13 pts**).