Software Engineering Methods Q1 2016

Instructor: Dr. Alberto Bacchelli Assignment 05

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 08

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 **Oct 24, 2016 @ 10:45AM**.
  - A sprint retrospective for this assignment, using the template "Sprint retrospective - template" available on Blackboard, by **Oct 28, 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 **Oct 28, 2016 @ 17:55**[2] and the commit that you TA has to consider must be tagged.

[1] *e.g.*, a correct name would be: *Group0-05.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 1 - Anonymous peer suggestions (30 pts)

Last week you were asked to understand, read, and analyze the code of another anonymous group. In addition, you had to propose meaningful enhancements to the other's group codebase, for a total of 30 points.

1. Ask your TA for the peer feedback that he received about your group and implement the proposed enhancements. Your TA will make sure that all the enhancements are relevant to you current codebase. (**30 pts**)

Exercise 2 - Software Metrics (45 pts)

The *inCode* tool uses software metrics to detect a number of design flaws. In this exercise you will use it to have guidelines to improve your implementation, from a code quality perspective.

1. Use inCode[3] to compute software metrics on your project, then upload the resulting analysis file[4] to your git repository. Write in the explanation PDF file where the analysis file is located (**3 pts**).
2. Consider the 'System summary' view (see Figure 1, Point 1) regarding the analysis of your project. You can see the design flaws that seem affect your system.[5] Pick the first **three** design flaws (in order of severity, see Figure 1, Point 2) that affect your software, and for **each** flaw complete the following points:
   a) Explain the design choices or errors leading to the detected design flaw (**4 pts**).
   b) Fix the design flaw or extensively and precisely explain why this detected flaw is not an error and, thus, should not be fixed (**10 pts**).

   If your project has less than three design flaws, congratulations! In this case, consider other design flaws (to reach a total of three with the previous ones) that inCode could detect, and explain in detail where each of these design flaws could have probably affected your system and how you managed to avoid it (**10 pts** per design flaw).

Exercise 3 - Teaming up (15 pts)

1. In this exercise, you decide *together* with your TA, during the group meeting on Monday, *new game features* to add to your game.[6]

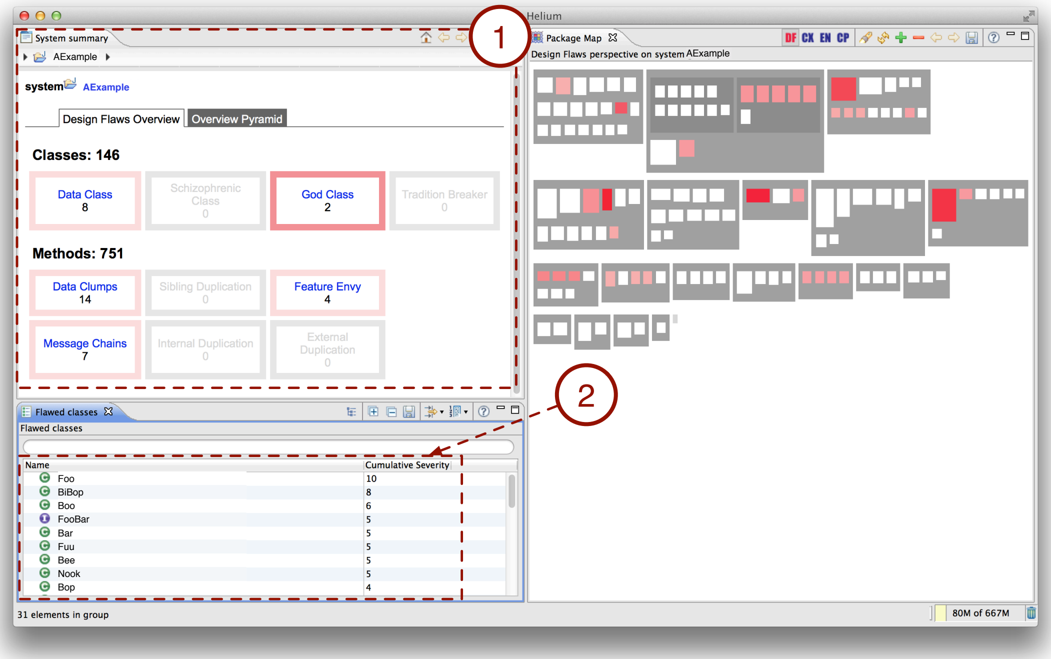
   After you decide on the tasks, write a requirements document, which will be evaluated in the same way as for the requirements document of the initial version. Afterwards you must implement the requirements. (**11 pts**).
2. During the analysis and design phases of this extension use responsibility driven design and UML (push to the repository the *single* PDF file including all the documents produced) (**4 pts**).

[3] Download the trial for Windows from: `https://drive.google.com/file/d/0BzuWZdqy9QYwZDhGM1U0SmxfZnc/view?usp=sharing`. This trial has sufficient features to complete this exercise.

[4] It can be found in the folder 'snapshots' in the inCode installation folder

[5] Remember: Software metrics only are half of the truth, so it is possible that detected flaws are not actual flaws.

[6] Both you and the TA will have to take into account that this exercise is worth one-sixth of this assignment



system AExample

Design Flaws Overview | Overview Pyramid

Classes: 146

Data Class 8	Schizophrenic Class 0	God Class 2	Tradition Breaker 0

Methods: 751

Data Clumps 14	Sibling Duplication 0	Feature Envy 4
Message Chains 7	Internal Duplication 0	External Duplication 0

Flawed classes

Name	Cumulative Severity
Foo	10
BiBop	8
Boo	6
FooBar	5
Bar	5
Fuu	5
Bee	5
Nook	5
Bop	4

31 elements in group

Figure 1: inCode analysis windows