HOGESCHOOL ROTTERDAM / CMI

Project 4

Developing a cross platform application

INFPRJ00-4

ECTS: 4
Module responsibles: M. Abbadi, L. Muilwijk

Description of the course

Modulenaam:	Project 4 – Developing a cross platform application
Modulecode:	**INFPRJ00-4**
Aantal studiepunten en studiebelastinguren:	This course provides you with four (4) study points, which corresponds to a workload of 112 hours. The recommended distribution of these 112 hours during the study weeks is as follows: <u>Supervised lectures:</u> Kick-off: 3 * 50 minutes 2,5 hours Project lesson (for 3 weeks): 6 * 50 minutes 15 hours Presentation of the product: 3 * 50 minutes 2,5 hours <u>Unsupervised hours:</u> Time to work on the project incl. literature study 92 hours **Total** **112 hours**
Vereiste voorkennis:	Basic programming and database knowledge is required.
Werkvorm:	Project-based education (group-work)
Toetsing:	Examination is based on the delivered product and the process of the project.
Leermiddelen:	Development tools, Dev4 literature, Anl4 literature
Draagt bij aan competentie:	▪ Advising (**A**dviseren) ▪ Design (**O**ntwerpen) ▪ Implementation (**R**ealiseren)
Leerdoelen:	▪ [**A1**] You can clearly and convincingly present your product to an audience with different (technical) backgrounds. ▪ [**O1**] You can represent the application and the design patterns used in it through UML. ▪ [**R1**] You can implement a mobile application. ▪ [**R2**] You can adapt a mobile application to different platforms via design patterns.
Inhoud:	You learn to work in a group context (*process*) and to realize a project assignment (*product*) for a client.
Opmerkingen:	Presence is **required**. Students make groups themselves. They have to communicate these to the tutor before the end of the first day of the project (through e-mail). Groups should be 4 or 5 students each. The tutor has to give approval and has the ability to make some adjustments if needed. *Note*: Groups that, during the course of the project, lose team members and remain with 3 (or less) students, will discuss with their project teachers about adjusted criteria for the evaluation.
Modulebeheerder:	M. Abbadi, L. Muilwijk
Datum:	27 May 2016

1. General information

1.1 Introduction

You've already taken part in 3 projects. You've gained substantial experience in working together in a project group and you've mastered the Scrum method. You might have been a Scrum Master once and you've had several opportunities to develop products using a new programming language. You delivered several (programmed) shippable products and have grown more competent in the working field of a software engineer. This project will focus on a new aspect of this field: visualizing data in a mobile application and trying to abstract it to a cross platform application by means of design patterns.

You will work in a project team again, in a 3 week time frame. Once again you'll be working together on a case that has to be finished in the end of week 10. Your Product Owner and Tutor will set the rules with regard to delivery and will guide you through the process. Feedback will be given regularly.

1.2 Relationship with other courses

You will need to apply all the knowledge learned during the three courses (Skl, Anl, Dev) in the project itself.

The knowledge acquired during the Skills course (INFSKL02-1) is tightly connected and applicable to every project you will encounter. This period Skills focuses on presentational Skills, as will be visible in the delivery criteria for the Tutor. The courses of Analysis INFANL01-3 and Development INFDEV01-4 are also crucial to bring this project to a satisfying end. In the ANL course you learned how to represent applications with UML. In the DEV course the focus was on design patterns.

1.3 Learning materials

Mandatory:

- N@tschool (for deliveries)
- Dataset provided on N@tschool
- Github, https://github.com/
- ***Technical requirements***
    - For the mobile version you must use Java or C# (or F# or Haskell, but only with consent from your P.O.).
    - In addition, for the desktop version you must use JavaFX or Monogame.

Facultative:

- Franken, M. (2013), *Scrum voor Dummies* (1e druk), Amsterdam: Pearson Education Benelux B.V., ISBN 978-90-430-2403-7
- Trello, https://trello.com
- Literature from Analysis 2, and 3, Development 2, 3, and 4
- https://www.scrumalliance.org/community/articles/2008/september/definition-of-done-a-reference

2. Program and contents

2.1 Assignment

The goal of both Projects 3 and 4 is to create an informative tool about the city and its neighbourhoods, for current/potential citizens of Rotterdam. The final product should be a multiplatform application (at least **mobile and desktop**) and it should contain visualizations/animations on interesting facts related to the city of Rotterdam.

During **Project 4** you will get datasets as CSV files from which you have to create and display some visualizations in a cross platform application environment. You have to focus on the creation of an architecture that is potentially able to support multiple platforms by reusing most of the shared logic code. Visualization depends on the platform. Reuse and flexibility are obtained through the application of design patterns. **Important note**: the *mobile version* of the application is required to pass the course, while the flexible architecture supporting also the *desktop version* gives additional points but is not required: see Attachment 3 for more details on the evaluation.

More **detailed** information:

- The data sets that you have to use are:
  - http://rotterdamopendata.nl/dataset/fietstrommels
  - http://rotterdamopendata.nl/dataset/fietsdiefstal-rotterdam-2011-tot-2013
- The application should contain the following features (see Attachment 3 for more information on the grading), sorted by Moscow priority:
  - [**M**] Show in a bar chart the 5 neighborhoods with the biggest amount of bike containers (*fietstrommels*).
  - [**M**] Show in a line chart the amount of stolen bicycles per month.
  - [**M**] Show the amount of stolen bicycles and amount of installed bike containers per month given a specific neighborhood through a grouped bar chart. The neighborhood should be selected by the user.
  - [**M**] Show in two pie charts the distribution of most stolen bike brands and colors.
  - [**M**; *only mobile version*] Save the current location of your bike in the note app. **Note**: your application must be capable of writing the note without explicitly opening the note app.
  - [**M**; *only mobile version*] Create a reminder/appointment in the agenda about when to pick up the bike from a specific location. **Note**: Your application must be capable of putting the appointment in the agenda without explicitly opening the agenda application.
  - [**S**] Show on a map the amount of thefts per neighbourhood, by coloring the neighbourhood differently (shades of red/orange/yellow/green) based on the thefts amount. The border between neighbourhoods must be (somehow) clear.
  - [**S**] Show the location and route indication to the closest bike container to the user, given his/her current position.
- Creating the database for the application is optional since you are using a CSV file directly as input.
- In each visualization the legenda, the title and the values on the axes should be indicated accurately.

2.2 Week scheme

The project covers the last three weeks of OP4: week 8 (sprint 1), week 9 (sprint 2) and week 10 (sprint 3). In this project it is *again* mandatory to use Scrum as a project method (see Attachment 1 for more details).

In the following table you can see the lessons of each week and the corresponding deliveries.
Deliveries must be done **no later than 12 hours** before the start of the corresponding lesson. For example, if your lesson is at 10.30 of Wednesday, then you must deliver before 22.30 of Tuesday.
The deliveries will be done through N@tschool[1].

[1] The scrum master of each group will have the duty of uploading the documents to deliver for his/her group in a compressed file called “INF1X – Group Y – Week Z”.

Week	Day (see schedule for detail)	Teachers present[2]	Topic	Deliveries (deadline: 12 hours before the lesson!)
8	Monday morning	P.O. and Tutor	Kickoff together with all the first-year students and with your class	/
8	~half of week	Tutor	Feedback on collaboration and project-management	Cooperation contract, DoD, Scrumboard[3], Demo P.O. review[4]
8	~end of week	P.O.	Review sprint 1	-- UML (draft 1) representing the the entire application emphasizing the design patterns used; -- Product Backlog and Sprint Backlog (for sprint 2); -- Shippable product [no upload needed; you must be able to show it during the review];
9	~half of week	Tutor	Feedback on collaboration and project-management	DoD, Scrumboard, Demo P.O. review
9	~end of week	P.O.	Review sprint 2	-- UML (draft 2) representing the the entire application emphasizing the design patterns used; -- Product Backlog and Sprint Backlog (for sprint 3); -- Shippable product [no upload needed; you must be able to show it during the review];
10	~start of week	P.O.	Feedback on progress (*on request*)	/
10	~half of week	Tutor	Feedback on collaboration and project-management	DoD, Scrumboard
10	~end of week	P.O. and Tutor	Final presentations and evaluation (review sprint 3)	-- Complete documentation (UML diagrams with explanations); -- Final version of code with comments; -- Screenshots of the application; -- Presentation slides -- Github contributors plot

Important note: for all deliveries, read the evaluation attachments for further details.

[2] If compatible with personal work schedules, Tutors are invited to attend the P.O.s lessons, and vice versa.
[3] If you use a digital Scrumboard, you must invite the Tutor.
[4] For example, PowerPoint presentation (if it was used), beforehand prepared questions/doubts for P.O., etc...

3. Evaluation

	Evaluated by	Evaluated through	Partial result
Collaboration and project management	Tutor	*Attachment 1* (Evaluation form Tutor)	- Individual result between +1 and -1 or No Go (for individual commitment) - Possible team penalty of -1
Intermediate deliveries	P.O.	*Attachment 2 (Evaluation form PO)*	Yellow cards
Final product	P.O. & Tutor	*Attachment 3* (Evaluation form Final Product)	10 points

- The final grade is the sum of the partial results, with a maximum of **10 points** and a minimum of **1**.
- The forms associated to each evaluation part are given in the attachments.

3.1 – Examples

The following partial grades:
- 0 from Attachment 1
- No yellow cards from Attachment 2
- 5 points from Attachment 3

bring to a final grade of 0 + 0 + 5 = **5** (onvoldoende) → herkansing

The following partial grades:
- -1 from Attachment 1
- Yellow card from Attachment 2
- 7 points from Attachment 3

bring to a final grade of -1 -0.5 + 7 = **5.5** (voldoende)

The following partial grades:
- **Individual No Go** from Attachment 1
- Yellow card from Attachment 2
- 6 points from Attachment 3

bring to a final grade of onvoldoende → herkansing

3.2 – Herkansing (resit)

In case of an insufficient grade (*onvoldoende*) for INFPRJ00-4, the following scheme applies:

- If you got an individual No Go, then you **cannot** repeat the project this year;
- If your group did not complete the “basic features” (see Attachment 3), then you **cannot** repeat the project this year;
- In all other cases, you have to implement individually at least one *additional feature* (chosen by the PO from a list of features specific for the herkansing) by the end of **week 11**. The final grade is 4.5 plus the amount of points obtained through your individually programmed additional features. Each individual additional feature has a corresponding value in points (0.5 or 1).

If you do not succeed at the resit (or if you cannot repeat the project this year), you will need to follow this course again during next school year.

Attachment 1 – Evaluation form [Tutor]

CLASS: INF1…, Group…

Attendance

Attendance will be checked during every lesson. Attendance is obligatory to get a grade for the project. Only when a valid reason is given to the Tutor or Product Owner prior to the lesson, an exception might be granted. When you haven't met the attendance standards you will receive a lower grade (see specifications further below) or an ND/No Go (meaning you haven't taken part in the course). The Tutor will write down the attendance and individual points in the following table:

	Week 8 attendance	Week 9 attendance	Week 10 presentation	Individual points (between + 1 and -1 / No Go)	Team penalty for missing boxes (0 / -1)
Name 1					
Name 2					
Name 3					
Name 4					
Name 5					

The Tutor also has the opportunity to give you a higher or lower grader for your individual commitment. The individual commitment will be graded between +1 and -1 or a No Go.

Reasons when a Tutor, eventually in accordance with the group, might decide to give a different individual grade:

+1: the student has done, at least during one week, extraordinarily good work. He or she has developed something outstanding (and done visibly more than his/her group members).

0: the student has done well, but hasn't done anything as outstanding enough to give extra points. The average group grade that is calculated in the end represents rudimentary the same level as the individual deserves.

-1: the student has done considerably less work during the project in comparison with the rest of the group. However, he or she has recovered and still has done a reasonable amount of work (the student is able to show what he or she has done). Several items were uploaded too late and the student wasn't present on several (formal) occasions.

No Go: the student has delivered considerably less work during the entire project or almost all weeks. He or she has been a burden to the group's development and can't really show the amount of work he or she has done individually. The student was absent for a substantial amount of time and therefore hasn't done enough work. Being absent for several days without compensatory behaviour afterwards might be a reason to get a No Go.

Below are the criteria for the different deliveries for the Tutor. **If you miss more boxes than indicated below, your team gets a penalty of -1.**

Cooperation contract (week 8) → for this category you aren't allowed to miss any boxes.

Check if the grade is satisfactory:

Criteria	Week 8
The document is complete, professional and neat.	
There are no language mistakes in Dutch.	
All parts (like: division of roles, contact specifications, meeting moments, communicational tools etc.) are specific enough.	
All rules and consequences are clearly specified.	
All students are aware of the rules.	

Notification:

_____

_____

Demonstration for PO reviews (week 8 and 9) → for this category you are allowed to miss 1 box in week 8, but no boxes in week 9.

Check if the grade is satisfactory:

Criteria	Week 8	Week 9
The group has prepared a clear demonstration (for example slides) for the Product Owner.		
The group has prepared and documented questions/doubts for the Product Owner.		
The prepared work is professional (a coherent whole, no spelling mistakes, easily understandable).		

Notification:

_____

_____

Definition of Done (week 8, 9 and 10) → for this category you are allowed to miss 2 boxes in week 8, 1 box in week 9, but no boxes in week 10.

Check if the grade is satisfactory:

Criteria	Week 8	Week 9	Week 10
All listed activities add verifiable value to the product.			
It is clearly visible that the DoD consists of different levels: features, sprints and release.			
It is clear when a feature will be accepted.			
The DoD is a flexible document (and is constantly under development). Differences between the original version and the adapted versions (for example because of feedback from a tutor, P.O. etc.) are clearly visible.			

Notification:

_____

_____

Scrumboard (weeks 8, 9 en 10) → for this category you are aren't allowed to miss any boxes.

Check if the grade is satisfactory:

Criteria	Week 8	Week 9	Week 10
All tasks in the Scrumboard are roughly evenly distributed between team members.			
All tasks are specific enough.			
A (up-to-date) Product Backlog and Sprint Backlog are present.			

Notification:

_____

_____

Attachment 2 - Evaluation form (Intermediate deliveries) [P.O.]

CLASS: INF1…., GROUP …

Sprint 1 review

CRITERIA	POINTS
• The draft of the UML class diagram (version 1) is delivered on time, representing the entire application emphasizing the design patterns that you plan to use • There is a shippable product shown during the review	YES => 0 points NO => Yellow card [5]

Notes on feedback given:

_____

_____

_____

_____

_____

_____

Sprint 2 review

CRITERIA	POINTS
• The improved draft of the UML class diagram (version 2) is delivered on time, representing the entire application emphasizing the design patterns that you plan to use o The feedback received in week 8 has been applied and is visible in the improved UML draft • There is a shippable product shown during the review o The feedback received in week 8 has been applied and the results are visible in the application	YES => 0 points NO => Yellow card

Notes on feedback given:

_____

_____

_____

_____

_____

_____

[5] See *Attachment 3* for consequences of yellow cards on the grade.

Attachment 3 – Evaluation form Final Product [PO & Tutor]

During the last review, each group presents its product. During the presentation you must show the *features* of your product clearly, so that the Product Owner can evaluate the final result. The quality of the presentation will be evaluated by the Tutor. The final grade for this part is obtained by summing up the points for each criteria.

CLASS: INF1…., GROUP …

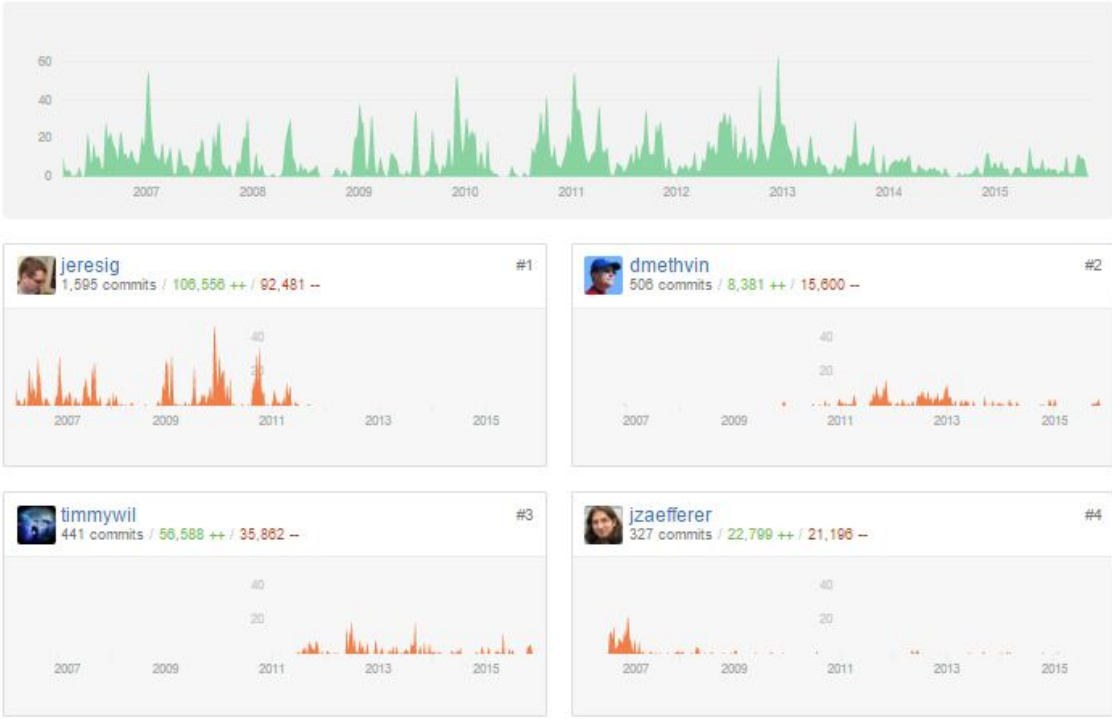
CRITERIA	POINTS
• The complete product is delivered on time	YES => 0 points NO => Yellow card
• The complete product contains: code, improved UML diagrams with explanation, contributors plot from Github, screenshots of the application, presentation slides • Code is commented • Each code file contains a maximum of 350 lines of code	**Prerequisite** to get the points of Week 10
The **mobile version** of the application contains the following ***basic features***: ● All Must-have features [M] (see paragraph 2.1) are implemented ● All features are reachable from the main menu of the application ● The application and the design patterns used are represented through UML diagrams ○ Automatic reverse engineering is not allowed! ○ Every UML diagram contains the mapping between the formal definition of the design pattern and the concrete application (see Attachment 5 for an example)	YES => 4.5 points NO => 0 points
The final **presentation** has the following features: ● Each member of the group contributes evenly to the presentation ● The presentation is complete (process review, demonstration of the product and UML class diagrams) ● The presentation is long enough (approx. 15 minutes) ● The presentation is understandable for everybody present (group members, fellow students, P.O. and Tutor) ● The presentation is well structured ● Verbal and non-verbal aspects are taken into account ● Appropriate visuals are used and explained properly	YES => 1 point PARTIALLY => 0.5 points NO => 0 points
The application includes the following ***additional features****:* ● The same logic code is reused for a *desktop application* through a different visualization layer; moreover, the UML documentation contains also the desktop part of the application [3.0p] ● *Should-have* (S) *features* have been implemented [0.5p each] ● There is a *database* used by the application [0.5p] ● *Free features* implemented in agreement with the P.O. [0.5p or 1.0p max] ● [only if using F#/Haskell] The design patterns used are functional in nature: ○ Monads [1.0p]	**NOTE**: If you don't get the points for the *basic features* or the teachers are not satisfied enough by their quality, no points for *additional features* will be given. Score of additional features: _____

○ Records of lambda's [1.0p] ○ Functors and natural transformations [1.0p]	
The group received yellow cards...	0 yellow cards => 0 points 1 yellow cards => -0.5 points 2 yellow cards => -1.5 points 3 yellow cards => -3 points

NOTE: In the Github contributions plot there should be no big difference in terms of contributions between each member and the rest of the group. If the contribution of a member is substantially lower than the contribution of the others, this will be considered as additional information by the Tutor to decide on the individual points (see *Attachment 1*).

Attachment 4 – Github contributors plot

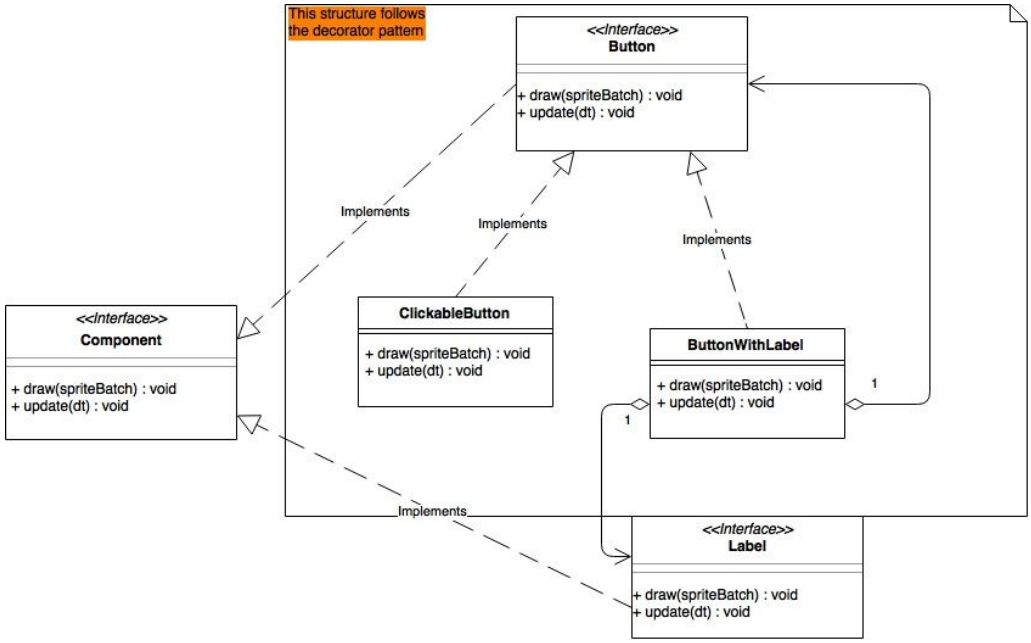
At the end of the project you must produce a screenshot of the "**Contributors**" graph that can be automatically generated from Github. Each member should briefly comment on his part of the plot (explaining his/her personal contribution).
More information on Github graphs can be found here. An example of the Contributors graph is the following:



Attachment 5 – Example on the mapping between formal design patterns and concrete implementation through UML

The following UML class diagram shows an example of a concrete implementation of the **decorator** design pattern and how you should show it for the deliveries to the P.O.. Note that the design pattern is enclosed in a rectangle to make it clearly identifiable.

Concrete implementation of a Decorator design pattern



The drawing must be accompanied by the generic version of the corresponding design pattern class diagram, like in the following picture. You must also underline the correspondence between the two diagrams, like this:

- the *Component* entity corresponds to the *Button* interface
- the *ConcreteComponent* entity corresponds to the *ClickableButton* class
- the *Decorator* and *ConcreteDecorator* entities correspond to the *ButtonWithLabel* class

Formal representation of a generic Decorator design pattern