AA 2017-2018 Software Engineering 2—Mandatory Project goal, schedule, and rules

READ THIS VERY CAREFULLY
NO EXCUSE FOR IGNORING WHAT WE WRITE HERE

Table of contents

1 Goal and approach

The objective of this project is to apply in practice what you learn during lectures with the purpose of becoming familiar with software engineering practices and able to address new software engineering issues in a rigorous way. The project includes two assignments:

1. The preparation of a Requirement Analysis and Specification Document (RASD) for a problem we provide you.
2. The definition of the Design Document (DD) for the system considered in point 1 above.

The two assignments will be reviewed during the final discussions that will take place during the winter exam sessions according to a schedule that will be proposed in the forthcoming months. The evaluation will assess the quality of the artifacts you prepare (accurateness, completeness, soundness) and the quality of your presentation (if you are able to explain your point in an appropriate way and if your presentation fits in the allowed time). Please check the introduction to the course for more information on the evaluation criteria for the mandatory project. The two assignments are described in the rest of this document.

2 Project schedule

- Group registration deadline 1/10/2017
- RASD submission deadline 29/10/2017
- DD submission deadline 26/11/2017
- Final presentation (to be scheduled)

All deadlines are assumed to expire at **23:59** of the days listed above.

3 Rules

- The project has to be developed in groups of two or three persons. Groups composed of a single student are allowed even if strongly discouraged. The assignments and the corresponding expectations of the professor will be calibrated based on the size of the group.
- Each group MUST register to the project following the steps indicated in Section 4. "Mixed" groups involving students of the two sections are allowed. When registering, each such group will need to indicate a single "reference" professor. This will be the one holding the discussion at the end of the course and deciding the grade. The choice is up to the students, but if we will realize that there is an unbalance between the groups under the responsibility of each of us, we may change your reference professor. In this case, we will inform you a few days after the group registration deadline.
- Each group MUST provide the requested artifacts within the stated deadlines. A delay of a few days, if notified in advance to the reference professor, will be tolerated but it will also result in a penalty in the final score. It is mandatory to provide these artifacts and to present them to the reference professor in a final meeting that will be scheduled later.
- The material presented in one artifact is not fixed in stone. You can (and are encouraged to) provide updates after the discussion sessions in the project lab or at any point before the discussion at the end of the course, if you think these are needed.
- During the development of the project each group will keep track of the number of hours each group member works toward the fulfillment of each deadline.
- For any question related to the project that could be interesting also for the other groups, please use the forum available on the Beep website. We will answer as promptly as possible.

4 Group registration

You should form your group and register it by going through the following steps:

1. Create a private repository for your project on Github (https://github.com). Your repository should be named by combining the names of all group members. For instance, BianchiRossiVerdi will be the name of the repository of the group composed of the students Tommaso Bianchi, Maria Rossi e Veronica Verdi. Make sure that all group members have a Github account and have access to the repository. Moreover, invite your reference professor (Github accounts dinitto and matteo-g-rossi) to access your repository (reading access is sufficient). We would like to see that you use the repository not only to upload the final versions of your deliverables but also to commit your intermediate versions. We would like to see commits performed by all group members.
2. Register your group by filling in the following form https://goo.gl/forms/ijL00pzQ2P2BHZZJ3. Do not forget to include in the form all relevant data!

5 The problem: *Travlendar+*

(adapted from: http://score-contest.org/2018/projects/travlendar.php)

Many endeavors require scheduling meetings at various locations all across a city or a region (e.g., around Lombardy), whether for work or personal reasons (e.g., meeting the CEO of a partner company, going to the gym, taking children to practice, etc.). The goal of this project is to create a

calendar-based application that: (i) automatically computes and accounts for travel time between appointments to make sure that the user is not late for appointments; and (ii) supports the user in his/her travels, for example by identifying the best mobility option (e.g., use the train from A to B and then the metro to C), buying public transportation tickets, or by locating the nearest bike of a bike sharing system.

Users can create meetings, and when meetings are created at locations that are unreachable in the allotted time, a warning is created. As mentioned, the application should also suggest travel means depending on the appointment (e.g., perhaps you bike to the office in the morning, but the bus is a better choice between a pair of afternoon meetings, and a car – either personal, or of a car-sharing system – is best to take children to practice) and the day (e.g., the app should suggest that you leave your home via car in the morning because meetings during a strike day will not be doable via public transportation; it could also take into account the weather forecast, and avoid biking during rainy days).

Travlendar+ should allow users to define various kinds of user preferences. It should support a multitude of travel means, including walking, biking (own or shared), public transportation (including taxis), driving (own or shared), etc. A particular user may globally activate or deactivate each travel means (e.g., a user who cannot drive would deactivate driving). A user should also be able to provide reasonable constraints on different travel means (e.g., walking distances should be less than a given distance, or public transportation should not be used after a given time of day). Users should also be able, if they wish to, to select combinations of transportation means that minimize carbon footprint.

Additional features could also be envisioned, for instance allowing a user to specify a flexible "lunch". For instance, a user could be able to specify that lunch must be possible every day between 11:30-2:30, and it must be at least half an hour long, but the specific timing is flexible. The app would then be sure to reserve at least 30 minutes for lunch each day. Similarly, other types of breaks might be scheduled in a customizable way (**Groups composed of one single student are not required to focus on this point**).

For groups composed of three students: Travlendar+ should also offer the possibility to arrange the trips of the user. For example, it should allow users to buy public transportation tickets when a bus or the metro should be taken (but also the possibility that the user has a day/week/season pass could be considered), or it should locate the nearest car or bike of a vehicle sharing system if that is the means of transport of choice.

Single-person groups are not required to focus on the flexible lunch problem nor on the trip arrangement.

Groups composed of two people should focus on all aspects of the project description, except the point concerning trip arrangement.

Groups composed of three people should focus on all aspects of the project description.

6 The documents to be created

Each document you produce will include the following elements:

- **A FRONT PAGE** that includes the project title, the version of the document, your names and the release date.
- **The TABLE OF CONTENTS** that includes the headers of all the first three levels headings in your document with the corresponding page number. At the beginning of this document you find a table of contents that you can use as an example. Since in this document there are no level three heading (e.g., 3.1.1), they are not part of the table of contents as well.

The specific characteristics each document should have are described in the next subsections.

6.1 Assignment 1 - RASD

The *Requirements analysis and specification document (RASD)* contains the description of the scenarios, the use cases that describe them, and the models describing requirements and specification for the problem under consideration. You are to use a suitable mix of natural language, UML, and Alloy. UML and Alloy MUST be part of the documentation. You must also show that you used the Alloy tool for analysis, by reporting the models you obtained by using it. Of course, the initial written problem statement we provide suffers from the typical drawbacks of natural language descriptions: it is informal, incomplete, uses different terms for the same concepts, and the like. You may choose to solve the incompleteness and ambiguity as you wish, but be careful to clearly document the choices you make and the corresponding rationale. You will also include in the document information on the number of hours each group member has worked towards the fulfillment of this deadline. As a reference structure for your document, you should refer to the one reported below that is derived from the one suggested by IEEE.
Please include in the document information about the effort spent by each group member for completing this document.

1. **INTRODUCTION**
   A. *Purpose*: here we include the goals of the project
   B. *Scope*: here we include an analysis of the world and of the shared phenomena
   C. *Definitions, Acronyms, Abbreviations*
   D. *Revision history*
   E. *Reference Documents*
   F. *Document Structure*
2. **OVERALL DESCRIPTION**
   A. *Product perspective*: here we include further details on the shared phenomena and a domain model (class diagrams and statecharts)
   B. *Product functions*: here we include the most important requirements
   C. *User characteristics*: here we include anything that is relevant to clarify their needs
   D. *Assumptions, dependencies and constraints*: here we include domain assumptions
3. **SPECIFIC REQUIREMENTS:** Here we include more details on all aspects in Section 2 if they can be useful for the development team.
   A. *External Interface Requirements*
      A.1 *User Interfaces*
      A.2 *Hardware Interfaces*
      A.3 *Software Interfaces*
      A.4 *Communication Interfaces*

B. *Functional Requirements*: Definition of use case diagrams, use cases and associated sequence/activity diagrams, and mapping on requirements
C. *Performance Requirements*
D. *Design Constraints*
    D.1 *Standards compliance*
    D.2 *Hardware limitations*
    D.3 *Any other constraint*
E. *Software System Attributes*
    E.1 *Reliability*
    E.2 *Availability*
    E.3 *Security*
    E.4 *Maintainability*
    E.5 *Portability*

4. **FORMAL ANALYSIS USING ALLOY:** in this section you will include your Alloy model. We require you to comment on the model by discussing the purpose of the model, what you can prove with it and why what you prove is important given the problem at hand. You are also required to show one or more worlds obtained by running your model.
5. **EFFORT SPENT:** In this section you will include information about the number of hours each group member has worked for this document.
6. **REFERENCES**

6.2 Assignment 2 - DD

The *Design document (DD)* must contain a functional description of the system, and any other view you find useful to provide. You should use all the UML diagrams you need to provide a full description of the system. Alloy may also be useful, although its use is not mandatory here. You will also include information on the number of hours each group member has worked towards the fulfillment of this deadline. As a reference structure for your document please refer to the following one:

1. **INTRODUCTION**
    A. *Purpose*
    B. *Scope*
    C. *Definitions, Acronyms, Abbreviations*
    D. *Revision history*
    E. *Reference Documents*
    F. *Document Structure*
2. **ARCHITECTURAL DESIGN**
    A. *Overview*: High-level components and their interaction
    C. *Component view*
    D. *Deployment view*
    E. *Runtime view*: You can use sequence diagrams to describe the way components interact to accomplish specific tasks typically related to your use cases
    F. *Component interfaces*

G. *Selected architectural styles and patterns*: Please explain which styles/patterns you used, why, and how

H. *Other design deci*sions

3. **ALGORITHM DESIGN**: Focus on the definition of the most relevant algorithmic part
4. **USER INTERFACE DESIGN**: Provide an overview on how the user interface(s) of your system will look like; if you have included this part in the RASD, you can simply refer to what you have already done, possibly, providing here some extensions if applicable.
5. **REQUIREMENTS TRACEABILITY**: Explain how the requirements you have defined in the RASD map to the design elements that you have defined in this document.
7. **IMPLEMENTATION, INTEGRATION AND TEST PLAN**: Identify here the order in which you plan to implement the subcomponents of your system and the order in which you plan to integrate such subcomponents and test the integration.
8. **EFFORT SPENT:** In this section you will include information about the number of hours each group member has worked for this document.
9. **REFERENCES**