A.Y. 2020-2021 Software Engineering 2
Requirement Engineering and Design 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 R&DD project. The two assignments are described in the rest of this document.

2 Project schedule

- Group registration deadline 16/10/2020
- RASD submission deadline 23/12/2020
- DD submission deadline 10/01/2021
- Final presentation (to be scheduled)

Note: **You can submit before the deadlines, if you want/need!**
All deadlines are assumed to expire at **23:59** of the days listed above.

3 Rules

- This assignment is optional and replaces part of the written exam of the Software Engineering 2 course.
- The project is 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 interested in taking the project must register itself 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. These artifacts will be presented to the reference professor in a final meeting that will be scheduled later.
- Each group MUST release artifacts by committing them into a specific folder of the github repository created for the project (see the following section).
- Each group MUST use the repository not only to upload the final versions of deliverables, but also to commit intermediate versions. We want to see commits performed by all group members. In the case a group wants to use collaborative writing tools (e.g., Google Doc) that keep track of individual contributions, the group will include in the readme file associated with the github repository the link to the online document, making sure that through that link the reference professor will be able to inspect the contributions to the document.
- The material included in your artifacts is not fixed in stone. You can (and are encouraged to) provide updates at any point before 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.
- This year, some groups (not necessarily all) will be assigned a tutor from industry. Tutors will provide advice concerning the quality of the produced documents and how these can be improved. The selection of the groups which will be assigned a tutor, and the assignment itself, will be done randomly, and the groups will be picked among those that will have indicated in their registration form an interest for this tutoring activity. Please notice that we expect that there will not be enough tutors for all groups. The assignment of groups to tutors will be communicated by the end of November. Students who will be assigned a tutor will invite him/her to the project repository and will schedule with him/her two meetings, one right after the RASD submission and the other right after the DD submission. They will then update their documents based on the received feedback. In the assessment of groups with tutors, the assessment criteria will be the same as for non-tutored groups, but we will take into account the status of the initial version of each document submitted to tutors, the received suggestions, and the improvements achieved after applying the changes.

4 Group registration and organization of your repository

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). Please note that, as students, you have the possibility to create a private Github repository for free. 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).
2. Register your group by filling in the following form https://forms.office.com/Pages/ResponsePage.aspx?id=K3EXCvNtXUKAjjCd8ope6yc_eXQo2JtIn6GJIJmJ9CdUNVJQNDBQMUxPOUI5T1VOOEk2N0pHMTBPQi4u. Do not forget to include in the form all relevant data!
3. Create a directory for each of the documents you will be working on.
4. Moreover, create a directory called DeliveryFolder where, by the due deadlines, you will commit the pdf version of your documents (name it RASD1.pdf or DD1.pdf, depending on the document you are releasing) plus any additional file you may want to include (e.g., the Alloy model and/or any UML model).
5. After the deadline for submission, should you need to update your document, you can commit in the same folder another pdf file with an increased version number, e.g., RASD2.pdf. The new file should include a section that describes the performed changes.

5 The problem: CLup – Customers Line-up

The coronavirus emergency has put a strain on society on many levels, due to many countries imposing lockdowns that allow people to exit their homes only for essential needs, and enforcing strict rules even when people are justified in going out (such as limiting the number of accesses to buildings and keeping a distance of at least one meter between people).

In particular, grocery shopping---one of the most essential needs---can become a challenge in the presence of such strict rules. Indeed, supermarkets need to restrict access to their stores to avoid having crowds inside, which typically results in long lines forming outside, which are themselves a source of hazards. In these trying times, people turn to technology, and in particular to software applications, to help navigate the challenges created by the imposed restrictions.

The goal of this project is to develop an easy-to-use application that, on the one side, allows store managers to regulate the influx of people in the building and, on the other side, saves people from having to line up and stand outside of stores for hours on end.

The application would work as a digital counterpart to the common situation where people who are in line for a service retrieve a number that gives their position in the queue. Naturally, physically retrieving a number forces people to first approach the building, and then wait in close proximity (though not in a line) until their number is called, which is a less than ideal situation in a lockdown situation. A software application, instead, could provide many improvements to the situation described above.

For example, it would allow customers to “line up” (i.e., retrieve a number) from their home, and then wait until their number is called (or is close to being called) to approach the store. In addition, the application could be used to generate QR codes that would be scanned upon entering the store, thus allowing store managers to monitor entrances. For the application to effectively work in practice, all

customers should use it to access the store, which has a number of consequences, including the following ones:

- The application should be very simple to use, as the range of users include all demographics (everyone needs to do grocery shopping).
- The lining up mechanism should be effective. There is a real risk that the approach does not work in the case the customer arrives to the grocery store after his/her number is called, or too early, as in this case we would get back into a physical line situation. This implies that the system should provide customers with a reasonably precise estimation of the waiting time and should alert them taking into account the time they need to get to the shop from the place they currently are.
- Fallback options should be available for people who do not have access to the required technology; for example, stores should also have the possibility to hand out "tickets" on the spot, thus acting as proxies for the customers.

In addition to managing lines in real-time, the application could also allow customers to "book" a visit to the supermarket. This feature would be similar to the booking of a slot for visiting, say, a museum/exhibition, but with important differences. In particular, whereas one can expect that the time that it takes to visit a museum is fairly uniform (and people would typically want to visit the whole museum/exhibition), the same is not true for visits to the supermarket. Hence, upon booking a visit, a customer might indicate also the approximate expected duration of the visit. Alternatively, for long-term customers, this time could be inferred by the system based on an analysis of the previous visits. The application might also allow users to indicate, if not the exact list of items that they intend to purchase, the categories of items that they intend to buy. This would allow the application to plan visits in a finer way, for example allowing more people in the store, if it knows that they are going to buy different things, hence they will occupy different spaces in the store when they visit (thus respecting the requirement that people keep enough distance between them).
Other features that the application might have include a suggestion of alternative slots (in the same day, or in different days) for visiting the store, to balance out the number of people in the store, the suggestion of different stores of the same chain (or even of different chains, if the application is chain-independent) if the preferred one is not available, or the periodic notification of available slots in a day/time range (these are other important differences with respect to museums/exhibitions, which are unique, and which are usually visited only once).

6 Project Scope

You are required to explore the above problem and define the corresponding RASD and DD. In particular:

- **Groups composed of a single student** will focus on the features for "lining up" at the store, including the features to monitor the accesses to the store and manage the line of customers accordingly.
- **Groups composed of two students** will address the points required for groups of one student, plus the "book a visit" feature.
- Finally, **groups of three students** will address the points required for groups of two student, plus the features to suggest alternatives and options to customers.

7 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.

7.1 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. Any Alloy model should be validated through the tool, by reporting the models obtained by using it and/or by showing the results of assertion checks. 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 scenarios and 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:** This section should include a brief presentation of the main objectives driving the formal modeling activity, as well as a description of the model itself, what can be proved with it, and why what is proved is important given the problem at hand. To show the soundness and correctness of the model, this section can show some worlds obtained by running it, and/or the results of the checks performed on meaningful assertions.
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**

7.2 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, but not mandatory. 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
B. *Component view*
C. *Deployment view*

D. *Runtime view*: You can use sequence diagrams to describe the way components interact to accomplish specific tasks typically related to your use cases
E. *Component interfaces*
F. *Selected architectural styles and patterns*: Please explain which styles/patterns you used, why, and how
G. *Other design deci*sions

3. **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.
4. **REQUIREMENTS TRACEABILITY**: Explain how the requirements you have defined in the RASD map to the design elements that you have defined in this document.
5. **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.
6. **EFFORT SPENT:** In this section you will include information about the number of hours each group member has worked for this document.
7. **REFERENCES**