RASD

Software Engineering 2

Antonio Ercolani - 10621728

Vittorio Fabris - 10562731

Riccardo Nannini - 10626268

Academic year 2020/2021

Contents

1 Introduction

1.1 Purpose

The *Requirement Analysis and Specification Document* (RASD) is a standardized document used by software engineers to communicate the most important aspects of a system to be developed to the project stakeholders. The RASD represents the starting point of the so called "waterfall life-cycle" of a software system, therefore it's easy to understand how delicate and strategical is the role it performs. Programmers, testers, designers and everyone involved in the development will refer to this paper and a mistake could potentially cost thousand hours of additional work.

What we said so far is the purpose of this document. This specific RASD is about an application that tries to face the crowding of supermarkets during the coronavirus emergency. In the following section we are going to explain the scope of that software system.

1.2 Scope

The scope of the software is to give the users the possibility to line up to enter and shop into a grocery stores.
The stores that have registered to this service generate and distribute tickets to line people correctly, letting them do the shopping only if the QR-code scanned on their ticket is valid.
CLup offers different kind of functionalities and features:

- **Basic:** The user can line up in different ways: through a physical ticket taken outside the store, or through the application, that is implemented in a way such that people of every age can use.
  If there are already too many people queuing outside the shop, the software doesn't allow to distribute new physical tickets.
  The software can integrate the information derived from the two sources, virtual and physical bookings, so that overcrowding is avoided in the neighborhood of the store and thanks to the QR ticket validations, store managers can monitor entrances.
  When the user leaves the store he must show his ticket to the QR-reader again, so that the system can know it and let the queue proceed, with another customer enter the shop.

- **Feature 1:** Users that line up from home concede the system to see where they are, using the GPS position of their device to identify their zone. In this way only nearby grocery stores are shown to the user.
  What's more, these category of users can book a visit at a specific time, if there is a free spot in the schedule organized by the system, otherwise other arrival times or grocery stores are proposed to the user.
  The system, knowing the position of the user, his distance from the store and the vehicle that he has chosen to use to reach it, will provide him a notification that in a certain moment he should leave to arrive on time at the grocery store. If the user arrives late, the system can reschedule a booking for him, always considering to avoid overcrowding and the time preferences of the customer, inserting him into the line again.

- **Feature 2:** Because the system needs to balance in the best way possible the number of people inside the store, the user that books online his visit is also asked which kind of products he's going to buy. In this way the system can manage people along the store's aisles, reducing the possible interaction and proximity of customers, distributing their access into the shop in different times if it is necessary.

- **Feature 3:** When the grocery store registers to the service, it can decide if he wants to save the information about the preferences of customers about the products they buy. In fact, as explained in the previous feature, the user lets the system know what he intends to buy. The grocery store could use these preferences in order to manage a plan for the refill of its shelves and always be ready to satisfy the needs of its customers, avoiding the risk of not having some products to sell, but ordering them from their providers in advance.

1.2.1 World phenomena

WP1	The customer arrives to the store
WP2	The customer shops in the store
WP3	The customer waits outside the store

1.2.2 Shared phenomena

Controlled by the **machine**

SP1	The customer is notified that his turn is coming
SP2	The store generates and sends a ticket to the user (?)

Controlled by the **world**

SP3	The customer enters the store scanning his QR code
SP4	The customer reserves a place in the queue of a given store using the application
SP5	The customer reserves a place in the queue of a given store using the ticket totem outside the store
SP6	The customer books a visit to the store
SP7	The customer leaves the store
SP8	The store receives a request to join its queue

1.2.3 Goals

G1	At any time the number of people in the store must not be higher than the store limit or the limit is passed but the customers are expected to be in different areas of the store
G2	The physical queue of people outside the store is reduced
G3	Allow users to ”line up” for a store from home
G4	Allow users to wait for their turn at home and receive an alert when their turn is coming
G5	Allow users to book a visit to a store
G6	Balance the number of people between different store or timeframes with suggestions
G7	Allow the store to monitor entries
G8	Handle the order of entries between the queue and the booked visits
G9	Collects information in order to build statistics and infer mean datas

1.3 Definitions, Acronyms, Abbreviations

In this section we explain the meaning of some technical terms used in the document.

QR CODE	A *Quick Response code* is a kind of bar-code, readable by machines to retrieve information
ASAP	An *As Soon As Possible* preference indicates that the customer wants to line up immediately, without booking a visit at a specific time to enter the store
UML	The *Unified Modeling Language* is a modeling language used to describe the design of a software system

1.4 Revision history

1.5 Reference Documents

1.6 Document Structure

This RASD document is structured over 6 main chapters, described below.

Chapter 1 starts with a brief description of the purpose of this document. Then the focus moves on the involved software system, for which is given an overall description of the goals, scope and related phenomena. The chapter ends with secondary sections about the description of technical terms and the history of this RASD document.

Chapter 2 let the reader to better understand the "world" of the analyzed software system. It begins with a UML and then other kind of diagrams that explain how the application interact with the environment around it. The following sections are about the requirements the system must satisfy, also with a focus on the specific users needs. Finally, in the last section, we list the domain assumption we did.

Chapter 3 goes deep into the topics described in the previous chapter. Begins with the description of the user interface and continue with use cases and sequence diagrams to show how the system behaves in different situations, in respect to the application requirements. Then you will find sections on other system constraints on performance and design. The chapter ends with few words on non-functional requirements.

Chapter 4 tries to describe the main aspects of the software system we have seen so far with an alloy model.

Chapters 5 and 6 contain, respectively, an outline of the effort spent in the project by each team mates and the document references.

2 Overall description

2.1 Product perspective

The CLup system is designed as a complitely new software application. The user side of the software is intended to be used as a mobile application while the store managers are going to interface with the software through a web interface.
In order to estimate the time needed for the user to reach the store, the application needs to interface with the smartphone's localization system and needs the user to indicate the type of transportation that he is going to use.
The system will handle client that are not using the application with tickets provided by a ticket totem located outside the store. This machine will act as an interface to the software application.
Customers using CLup are going to be notified when their turn is coming in order to approach the store while those with a ticket outside the store will monitor a screen extern to the shop in order to know when it's their turn to enter.
In order to access the store, every customer is required to scan their QR code (either from the application or from the physical ticket) when entering and exiting the store: in both cases the system will activate an automatic door.
On the other side of the system, store managers are going to be able to monitor and manage the queue of their store as well as programmed visits.
The below class diagram models the application domain in order to give an overlook on the enviroment where the application is going to work.

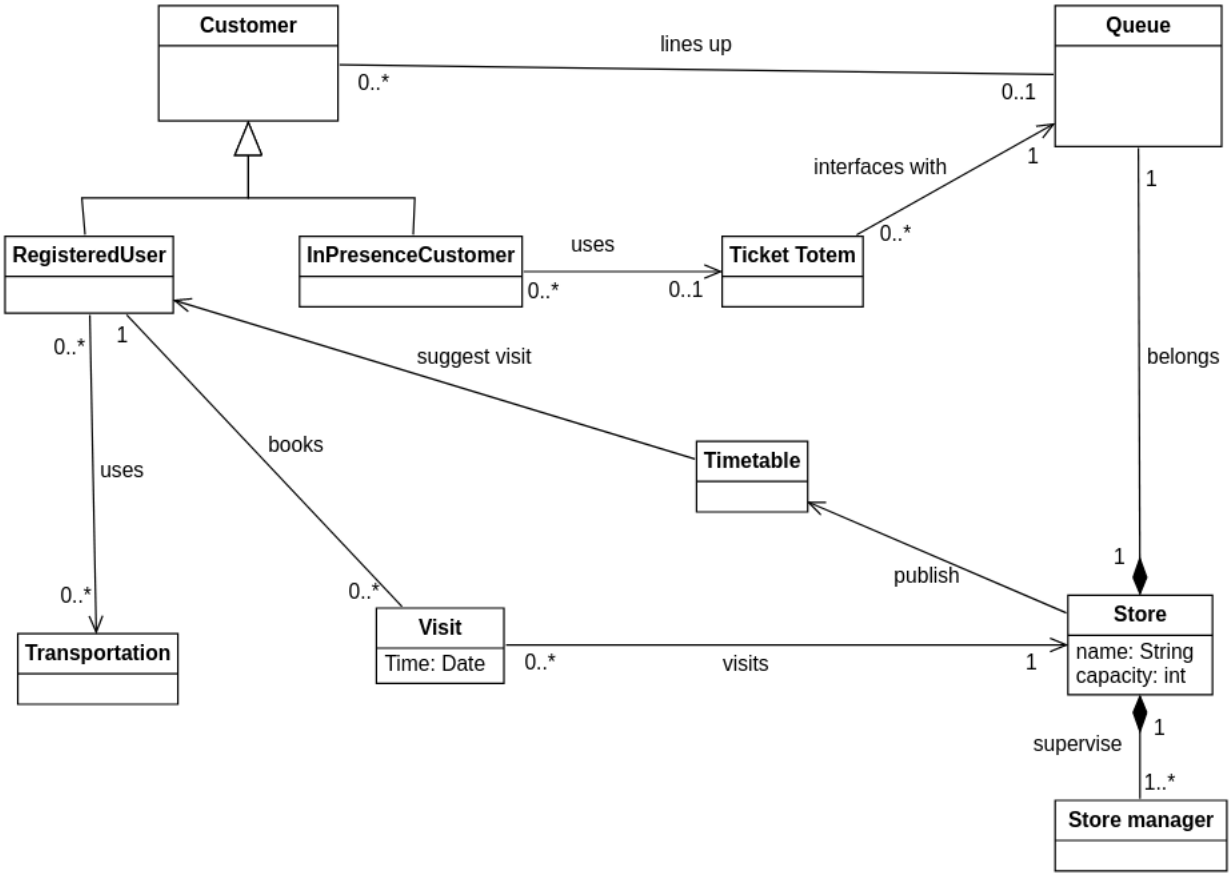

Figure 1: Class diagram

2.2 State Diagrams

In this section we want to show the evolution over time of the states of the application and the behavior of the user interacting with the app and the environment. The state diagrams below better explain the progress of the system.

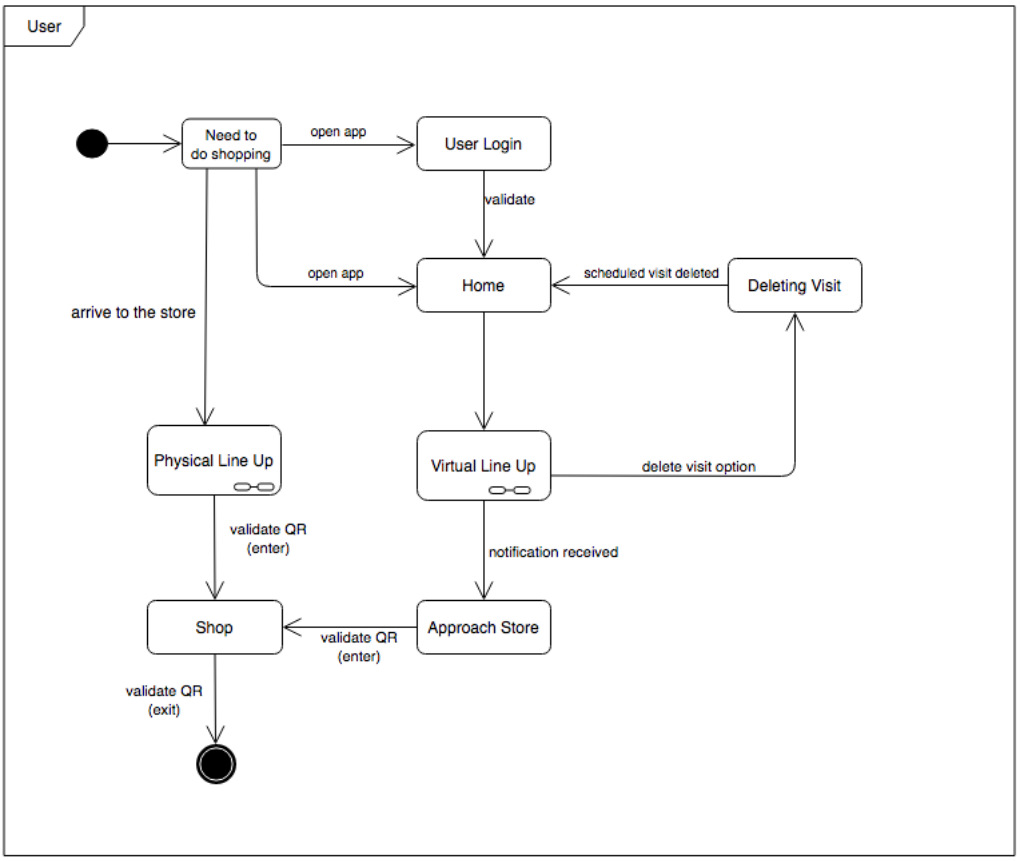

Figure 2: User State Diagram

In the first state diagram (Figure 2) it's shown how the user interacts with the environment and the system when he needs to do shopping. He interacts with the system lining up to a store in a virtual way using the mobile or web application, or queuing physically at the store he wants to visit. What's more, to enter the store, he needs to show the QR-code, and then he has to do the same when he exits, so that the queue can proceed.

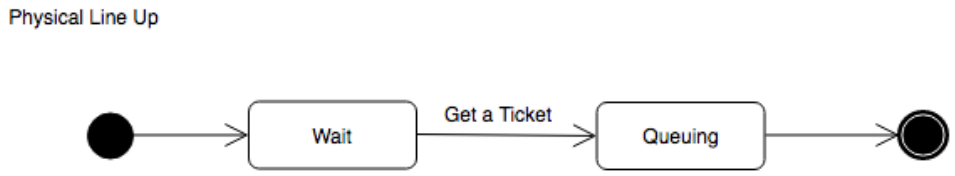

Figure 3: Physical Line Up State Diagram

In this second state diagram (Figure 3) it's better explained the physical line up of the user, that waits and directly gets a ticket from the totem nearby the store. The user then waits for his turn to enter and do the shopping.

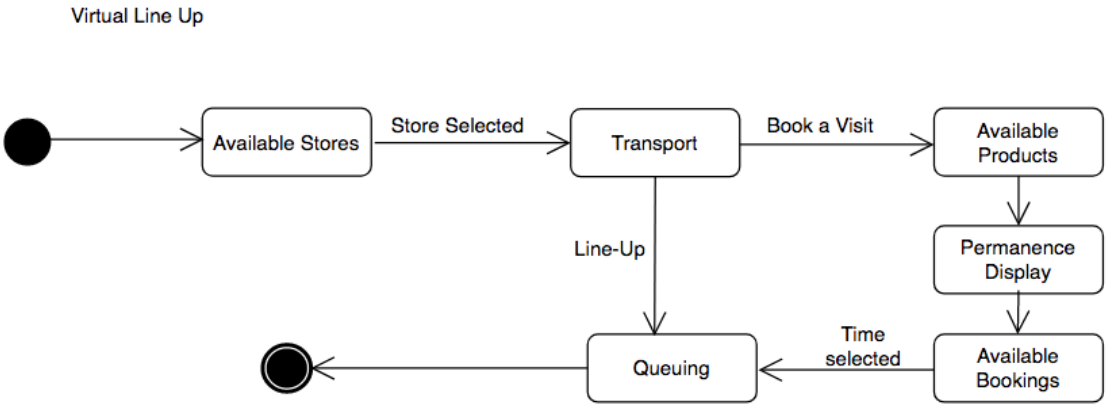

Figure 4: Virtual Line Up State Diagram

Here (Figure 4) are shown the steps to go through the virtual line up of the user, that has to face some intermediate steps before gaining the possibility to line up to the store. When the store has been selected, the user needs to indicate the transportation he will use to reach the store. Then, if he doesn't want to line up immediately but wants to book a visit, he will see the available products that he can buy and will have to choose which one of them he wants to buy, so that the system can collect his preferences. Then the user is asked to indicate his permanence into the store, in order to better manage the queue and the next available spots in the timeline.

2.3 Product functions

In this section are analyzed the main functionalities that the CLup software system must ensure. Goals and requirements are many and they are all connected with each other, therefore, below you will find the description of two "macro-functionalities" that try to group most of them. Said that, in order to go deeply in every system capability, the reader should supplement this section with the next chapter - "Specific requirements".

Queue management

The core function. As the name suggest, the system is able to manage the stores entering queue, that are becoming longer and "dangerous" in the time of coronavirus. This is done with a combination of two sub-functionalities:

- **Line up from home** - allow users to reserve their place in the queue without having to reach the store, they just need to access the application and choose a grocery. To this simple user action corresponds an important data elaboration made by the system, to ensure another important functionality: **alert the costumer when his turn comes**. In fact the system must take into account different factors to alert the user in the right moment:
  - The distance between the user home and the selected store;
  - How much time take people before the user in the queue to enter the store.

  Once the system has collected all the information, it can compute the right amount of time needed by the costumer to reach the store in the right moment he have to enter in.
- **Line up physically** - the system must give the possibility to line up physically. Therefore, when costumers reserve their place directly at the store the system must collect also this data. Obviously, for them there is no need to compute any kind of time to enter in the store because they wait for their turn right outside.

Booking visits

The second macro-functionality is the possibility of **booking a visit** to a registered store. On the costumers side, this functionality could be similar to the previous one but the main difference lies on the side of the stores. In fact the information that the costumers provide to the system during the booking is essential to the next presented feature.

- **Increase the capacity of the stores** - the system should allow more people in the stores than the standard number. This is made possible by asking to the costumer who book a visit the categories of items he's intended to buy, therefore the system can infer which sectors of the store he's going to occupy. These information are exploited by the system to maximize the capacity of the stores, respecting the social distancing imposed by the coronavirus emergency rules.

The second feature of the list is also directly linked to the "booking".

- **Balance out the number of people in the store** - the system should be able to balance out the number of people in the different registered stores. This is done by suggesting people alternative time slots or different stores when they are booking a visit.

2.4 User characteristics

The CLup software system involves category of users:

- **Costumers** - are client of the registered stores that want to take advantage of the CLup features. They are interested in the possibility of "line up" from home and in "booking visits", that encourage them to continue go shopping, despite the problem caused by the emergency.
- **"Stores"** - these users are represented by the stores that want to take advantage of the CLup services. They have to follow a registration procedure to be added to the available stores of the system.
  Besides exploit the "remote lining up" function that mainly face social distancing, they are interested in other features such as "increase the capacity of the stores" and "balance out the number of people in the store". In fact exploiting them they could are able to keep profits high, despite the coronavirus emergency.

3 Specific requirements

3.1 Use Cases

1. **Book a Visit**

Name	Book a Visit
Actor	User
Entry Conditions	The system is showing the available actions of the selected store and the user selects the "Book a Visit" option
Events Flow	1. The user indicates what kind of products he's going to buy 2. The user says how much time he will spend shopping 3. The system stores the user's preferences 4. The system displays the user a timetable with the available slots to enter the store, taking into account the avg time the user wants to spend inside the store and the products he wants to buy 5. The user selects one of the available time-slots 6. The system rearranges the queue inserting the user in it, reducing the available spots in the timetable 7. The system generates the virtual ticket with the QR code for the user
Exit Conditions	The system makes the ticket visible to the user
Exceptions	None

2. **Delete Visit**

Name	Delete Visit
Actor	User
Entry Conditions	1. The user has booked a visit 2. The user is still queuing
Events Flow	1. The user selects the delete visit option 2. The system displays the user the list of booked visits 3. The user selects the visits he wants to delete 4. The system deletes the scheduled visit and all the related data, updating the queue of the store 5. The user saves his operation
Exit Conditions	The user has successfully deleted his scheduled visit and the system has correctly upd
Exceptions	None

3. **Physical Queuing**

Name	Physical Queuing
Actor	User, Manager (for the Exception)
Entry Conditions	The user is arrived at the store
Events Flow	1. The user makes a ticket request at the totem outside the store 2. The system updates the queue, inserting another customer in the first available timetable slot 3. The user takes his ticket with the QR code
Exit Conditions	The user has the ticket to enter the store and can wait without standing near other p
Exceptions	If the totem is broken then the store manager will directly handle the physical reques

4. **Customer enters/exits the store**

Name	Customer enters/exits the store
Actor	User
Entry Conditions	The user is queuing from home
Events Flow	1. The user gets a notification to approach the store 2. The user arrives at the store 3. The user validates his ticket at the entrance through the QR reader because his turn has come 4. The system updates the queue 5. The user is doing shopping 6. The user validates his ticket at the exit of the store through the QR reader 7. The system updates the queue (AND SENDS A FINAL NOTIFICATION TO THE USER?????)
Exit Conditions	The system has correctly updated the queue of the store and is ready to allow a new
Exceptions	If the customer arrives at the store and his ticket is not valid anymore to enter the st

5. **Build Statistics / Preferences**

Name	Build Statistics / Preferences
Actor	User
Entry Conditions	The user has booked a visit at least 10 times and the system has stored his preference
Events Flow	1. The user selects the products he wants to buy and indicates his permanence in the store 2. The system collects the data, elaborates them and builds statistics 3. The results are shown to the user
Exit Conditions	The user can see what are his favourite products and the average time he usually spe
Exceptions	None

3.2 Software System Attributes - Non-Functional Requirements

3.2.1 Reliability and Availability

The system should be up at least for the 99,9% of the time, so the average time between it goes down and then it is recovered (MTTR) should not be higher than 9 hours per year. This also implies that the system should always guarantee that the needed services and functions are satisfied.
This high availability of the system needs to be provided because it is related to an emergency situation, and the users should always be able to decide to go shopping when they need (also because in this way they are likely to be distributed during the day). If this condition is not satisfied, there's the high risk that when the service will be recovered, all the people would go at the same time to the stores, and the overcrowding problem won't be avoided anymore.
Because we want to ensure this high grade of availability, a system of redundant servers can be considered. However, we can also take into account to have a prepared team that can recover the system as fast as possible when it's needed, while the users have been notified both when the system is down and when its services are restored and work properly.

3.2.2 Security

Passwords of the users and of the managers in the stores need to be protected, so before storing them they need to be correctly encrypted. A similar treatment has to be made also for the preferences pointed out by the users and stored by the system when they are lining up. What's more, for what concerns the managers, a periodical update of their password can guarantee a higher protection of the data collected and used by the store. Of course, the system should be protected against intrusion from agent that are not authorized to access it.

3.2.3 Maintainability

As previously said, the system must be flexible and easy to maintain. For this reason a proper use of design pattern has to be made, as well as a good level of abstraction, so that if the code needs to be extended with new features, everything could be made without any kind of deep code changes. Code must also be well commented and tests on main features of the system should always be working.

3.2.4 Portability

The software must run on different platforms, being compatible with the different devices that can be used: for example, managers at the store and users at home could use computers, so the services of the system need to be made usable on Windows OS, Linux OS and MacOS; because of the mobile application, it must be supported a version for Android, iOS and HarmonyOS.

3.2.5 Scalability

Everyone will use Clup because of emergency reasons, so the scalability of the system is one of the most important issues to deal with. An high reaction time needs to be reached, so as a consequence it must be set a proper number of processors, devices and memory.

4 Formal analysis using Alloy

5 Effort spent

Vittorio Fabris:

Discussion on the first section	**2h**
Scope	**3h**
State Diagrams	**3h**
Discussion on Requirements and Use Cases	**3h**

Riccardo Nannini:

Discussion on the first section	**2h**
World and Shared Phenomena & Goals	**3h**
Product perspective & class diagram	**3h**

6 References