

RASD - Requirement Analysis and Specification Document

Version 1.0 - 10-10-1019

Computer Science and Engineering – Software Engineering 2

Politecnico di Milano

Amedeo Carrioli – 866256, Andrea Ceruti – 945604

Professor Elisabetta Di Nitto

Table of contents

[Table of contents 1](#_Toc23498688)

[1. Introduction 3](#_Toc23498689)

[1.1 Purpose 3](#_Toc23498690)

[1.1.1 General Purpose 3](#_Toc23498691)

[1.1.2 Goals 4](#_Toc23498692)

[1.2 Scope 4](#_Toc23498693)

[1.3 Definitions, acronyms, abbreviations 6](#_Toc23498694)

[1.3.1 Definitions 6](#_Toc23498695)

[1.3.2 Acronyms 6](#_Toc23498696)

[1.3.3 Abbreviations 6](#_Toc23498697)

[1.4 Revision history 6](#_Toc23498698)

[1.5 Reference documents 6](#_Toc23498699)

[1.6 Document structure 7](#_Toc23498700)

[2. Overall Description 7](#_Toc23498701)

[2.1 Product perspective 8](#_Toc23498702)

[2.2 Product functions 11](#_Toc23498703)

[2.2.1 Reporting system functionalities management 11](#_Toc23498704)

[2.2.2 Map functionalities management 11](#_Toc23498705)

[2.2.3 Authorities management 11](#_Toc23498706)

[2.3 User characteristics 11](#_Toc23498707)

[2.4 Assumptions, dependencies and constraints 11](#_Toc23498708)

[3. Specific Requirements 11](#_Toc23498709)

[3.1 External interface requirements 11](#_Toc23498710)

[3.1.1 User interfaces 11](#_Toc23498711)

[3.1.2 Hardware interfaces 11](#_Toc23498712)

[3.1.3 Software interfaces 11](#_Toc23498713)

[3.2 Functional Requirements 11](#_Toc23498714)

[3.3 Performance Requirements 12](#_Toc23498715)

[3.4 Design constraints 12](#_Toc23498716)

[3.5 Software system attributes 12](#_Toc23498717)

[3.5.1 Reliability 12](#_Toc23498718)

[3.5.2 Availability 12](#_Toc23498719)

[3.5.3 Security 12](#_Toc23498720)

[3.5.4 Maintainability 12](#_Toc23498721)

[3.5.5 Compatibility 12](#_Toc23498722)

[4. Formal analysis using Alloy 12](#_Toc23498723)

[5. Effort spent 12](#_Toc23498724)

1. Introduction
   1. Purpose
      1. General Purpose

This document represents the Requirement Analysis and Specification Document (RASD). In this document we will explain SafeStreets. This will be done by a detailed presentation of the proposed solution and its purpose, listing its goals, and the requirements and assumptions through which they will be achieved.

SafeStreets is a public interface aimed to public-spirit citizens who want to help keeping the streets clear. This S2B intends to provide users with the possibility to notify authorities when traffic violations occur. This materializes using a platform through which users can upload pictures of streets violations, in particular parking violations.

There could also be other types of issues that a customer can report, for example speed violations, accidents, non-respected traffic lights. They’re reported in different ways.

The S2B also has a map, based on Google Maps, on which some areas are highlighted with different colors according to the number and types of violations reported (for example first the user can choose the type of violation, then the map shows different areas with different colors: a red area means that a lot of the chosen violation have occurred, yellow is medium quantity, green one means very few).

Once the violation is sent, its data are stored in SafeStreets center and analyzed by the software, in order to retrieve information to update the map.

The customers of the application are both singular users and authorities, for example the Police Department, that can find the S2B useful in order to maintain the public order.

An important point is that a user can eventually report fake violations. First, the application allows the user to report a violation even if the user’s geographical position and the violation’s positions are different (for example the user sees an illegal parked car while he is jogging but doesn’t want to stop his run to make the signalization, so he updates the violation once he gets home. This means that he could give a non-accurate position of the illegal parking). Secondly, there could actually be some users that find funny to report wrong violations, for now there is nothing we can do to fix this.

Finally, SafeStreets wants to offer a service exploiting the information of the municipality, if it allows users to retrieve the required information. The application crosses the information given by the municipality (only accident info), which are reliable, and the ones given by the users, which are not. Then it updates the map and make suggestions regarding possible solutions to prevent violations (only in red areas).

* + 1. Goals

-[G1] The application must allow users to send reports of streets violations. In particular, users can send reports even if they aren’t in the position where the infraction occurred.

-[G2] The application will have to store the information about violations and complete them with suitable metadata.

-[G3] The application must allow both end users and authorities to mine the information stored. This is done by coloring the map based on the number of violations.

-[G4] The system must update the map after every reporting.

-[G5] The system must be able to cross information received from municipality with its own data.

-[G6] The system must suggest to municipality possible interventions to prevent accidents.

-[G7] The system must send suggestions to authorities. When an area is red for a specific type of violation a suggestion is made.

* 1. Scope

As already mentioned, the SafeStreets system is made to provide users a service to report streets violations and a map where is possible to watch the areas in a safeness key.

Once the users sign in with a proper username and password, are in the system and can use all the services offered by the application.

As first, the system will allow to choose between looking at the map or report a street infringement. In the first case, it will be asked if the user wants to share the GPS signal with the application, to be more precise about the areas close to him. As mentioned before, on the map are highlighted different areas with different colors, based on the frequency of violations. Then the user can interact with the map. If a highlighted area is typed on, the number of infringements will be shown.

It is also possible to select the type of violation and the interval of time (today, last week, last month) in which the user is interested in watching the map. By default, if the user doesn’t select any type of violation or interval of time, the map of all types of violations together occurred in the last month will be shown.

Regarding to the reporting of violations, the systems allows to select the type of reporting (parking, speed, traffic light violation or accident), only in case of parking is possible to send a picture. The system will ask for a picture with visible license plate, if the user doesn’t send one, he can type the license plate by himself (not mandatory).

When parking violation is selected, it is also possible to choose the specific type of parking. The possible choices are:

* Crosswalk
* Sidewalk
* Double-parking
* Street crossing
* Red zone
* Taxi zone
* Bus zone
* Handicap spot
* Other

SafeStreets has a software that recognizes the areas in which a specific type of parking violation is above a certain number, this means that a lot of the same type of infringement occurred in the same area. A suggestion is made by the software, based on the type of parking violation. For example, if there are many sidewalk violations, the suggestion made will be “add a barrier between the street and the sidewalk to prevent unsafe parking”. These suggestions are sent to authorities.

After any reporting, the user is asked if he wants to keep anonymous his reporting or not. If he chooses to be anonymous, on the map won’t be shown his username along with his reporting, otherwise his username will be shown.

To prevent any misuse of the reporting violation system, if the user sends a picture with visible license plate and the application analyzing pictures algorithm fulfills in recognizing it, then that reporting will be shared with authorities. Otherwise, either if the user only types the license plate without sending a picture of it or the application analyzing pictures algorithm can’t recognize the plate, it is considered not reliable, so the reporting won’t be shared with authorities. It will only be added on the map.

For speed infraction, traffic light violation or accident reporting, is not possible to send a picture, in order to prevent the use of phones while driving. The user can only choose the type of violation.

The customer has also to choose the time and geographical position of the infraction, which is independent from the type of violation. It’s either possible to select the user’s position (using the GPS) or selecting a position on the map or typing the address.

The user can’t open the gallery on his smartphone to update a picture. This is for preventing the update of fake parking pictures that were shot in a time we can’t know. The user can open the camera through SafeStreets application and, either take the picture and send it, or take a picture and, if he doesn’t have time to send it, the S2B puts a timer of 2 hours on the photo, during which the user can re-open the application and upload the picture. If the timer expires the picture gets deleted.

If the municipality offers a service through which is possible to retrieve information about the accidents, SafeStreets analyzes them and crosses them with the SafeStreets’s ones. This is done in order to updated the map, which means identify unsafe areas (SafeStreets sees the municipality information as they were reliable reportings) and to give suggestions to the municipality itself to prevent more violations (for example add barrier between the bike lane and the part of the road for motorized vehicles to prevent unsafe parking).

* 1. Definitions, acronyms, abbreviations
     1. Definitions
* User: the costumer of the application that send reports. It could be a private citizen or an authority like municipality. In this case the use of the application will be different.
* Application analyzing pictures algorithm: the algorithm that SafeStreets uses for recognizing the license plate of the car object of the violation.
  + 1. Acronyms
* RASD: Requirement Analysis and Specification Document.
* API: Application Programming Interface.
* GPS: Global Positioning System.
* S2B: Software To Be.
  + 1. Abbreviations
* Gn: nth goal.
* Rn: nth requirement.
* Dn: nth domain assumption.
  1. Revision history
* Version 1.0:
  + First release.
  1. Reference documents
* Specification document: “SafeStreets Mandatory Project Assignment”.
* IEEE Std 830-­‐1998 IEEE Recommended Practice for Software Requirements Specifications.
* Examples documents:
  + RASD Sample from A.Y. 2015-2016.pdf
  + RASD Sample from A.Y. 2016-2017.pdf
  1. Document structure

The RASD is composed by 5 sections.

* Section 1:

it is the introduction of the RASD in which the problem is presented informally with natural language. It provides base information such as the product to develop and the application domain. The scope part is an analysis of the world and the shared phenomena.

* Section 2:

it presents an overall description of the project. It describes external interfaces, summary of major functions, constraints, assumption and dependencies of the S2B. Furthermore, a class diagram and some state diagrams are provided to make stakeholders better understand the project, but even for giving more details on shared phenomena and the domain model.

* Section 3:

this is the body of the document. It first describes the interfaces requirements. Then it lists some scenario to show how system works in real life situations and functional requirements. Lastly, we have nonfunctional requirements such as performance requirements and design constraints. This section will be useful for the development team.

* Section 4:

here we have the Alloy formal description of the problem that includes all the relevant details.

* Section 5:

it presents the effort spent for every member of the group.

1. Overall Description
   1. Product perspective

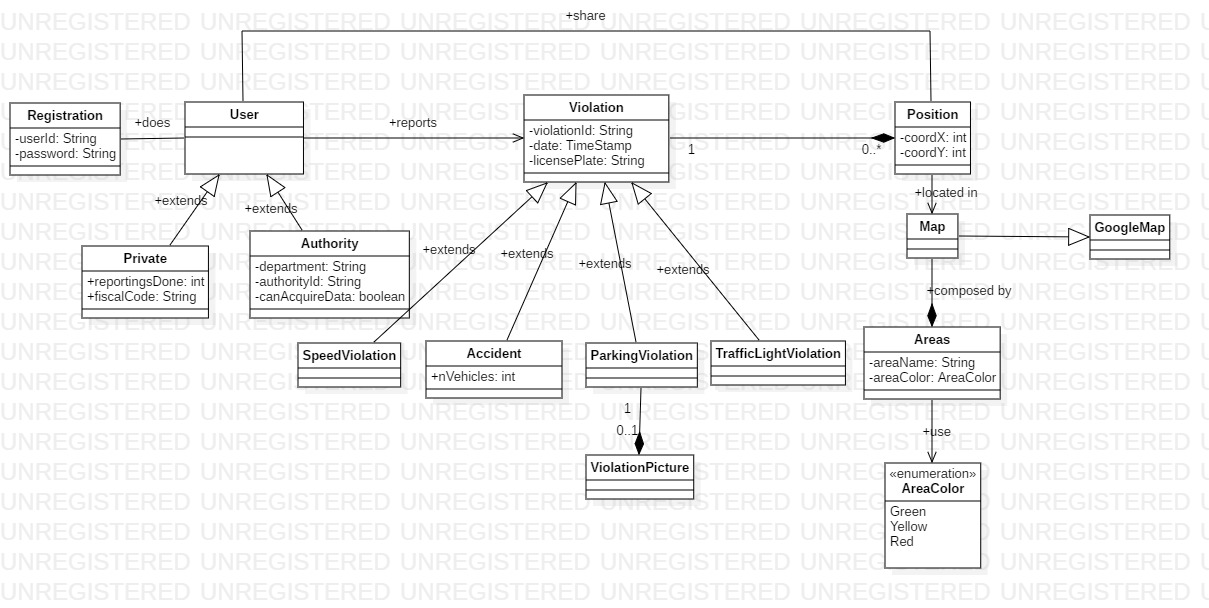
The SafeStreets system is a completely new software application, designed from scratch. It is intended to be used as a mobile application.

The S2B receives reporting of accidents and parking, speed and traffic lights violation. It stores the information on the application data center. Then updates the map every time a reporting occurs. The map as already said is based on Google Maps’ APIs because it offers a very large library of APIs.

The application also works along with authorities. Authorities can only acquire reliable data from SafeStreets. A reporting is reliable only if it is a parking violation reporting, with a picture, and the algorithm to read the license plate fulfils.

Anonymity is an important aspect that must be respected. When authorities acquire data, the username of the costumer that made the reporting won’t be sent. Only the reporting along with the picture, geographical position and time will be sent.

The municipality can also send information about accidents to SafeStreets. The information will be used as they were reliable reportings.



*Figure 1 - Class Diagram*

The class diagram (figure 1) shows the front-end side of the application. Users can be either private costumers or authorities, both of them must make a registration. Each private costumer has an attribute that stores the number of reportings that he has done.

A user can make a reporting violation. The violation will have a violationID, a date and a license plate, which is not mandatory. A violation occurs in one position, and in one position can occur many violations.

Only with a parking violation a picture can be sent, in the other cases the probability of the user is driving while noticing a violation is very high (almost every accident is sees while driving, for example on the highways. Also speed violation: while driving it’s easy to notice a car that speeds, for example if it overtakes you. Taking a picture while driving is very risky, besides being illegal).

A user can share the position of an infringement (using GPS, if he allows to). All the violations’s geographical position are shown on the map, grouped in areas, that are differently colored (green, yellow, red) based on the frequency of the violations.

Immagine che contiene testo, mappa

Descrizione generata automaticamente

*Figure 2 - State diagram 1: parking violation*

This first state diagram (Figure 2) shows how the application works on a parking violation reporting. The user chooses the type of violation, then inserts the information about it.

Once it is sent to SafeStreets, the data is stored, then if a picture is present, the license plate recognizing algorithm is ran. If the reporting is reliable (algorithm fulfills) the data are sent to authorities, otherwise it is only used to update the map.

Immagine che contiene testo

Descrizione generata automaticamente

*Figure 3 - State diagram 2: map usage*

Figure 3 describes how the map works along with selection of the users. Firstly, the map shows all types of violations together which occurred since a month ago from now. Then, the user can choose a specific type of violation, parking violations, which occurred in a chosen time interval, last week.

Now the areas on the map regards parking violations that occurred last week, and the user can select by typing on it a colored area, in this example a red one. The area shows the exact positions of the violations (the position selected by the costumer that sent it) and the number of them. The user can type on a specific violation, information about it will be shown, including the date and time of the reporting, along with the picture if present.

As already said, after every reporting, the user is asked if wants the reporting to be an anonymous or not. If he chooses it to be anonymous, his username won’t be shown to other users that select his violation on the map.

Immagine che contiene screenshot

Descrizione generata automaticamente

*Figure 4 - State diagram 3: behavior on municipality info*

State diagram 3 shows how SafeStreets works on the information the municipality shares. Information are stored on SafeStreets data center and considered reliable information. Then the map is updated with the violations arrived from the municipality. The reportings will be anonymous and they won’t have a picture. The geographical position is given by the municipality, along with the time and the license plate can either be sent or not.

* 1. **Product functions**

In this section will be analyzed all the funcions that SafeStreets offers. Some have already been described in the previous parts. Here are listed and more precisely specified, with respect of the already mentioned goals of the system.

* + 1. **Reporting system management**

This is the main functionality of the SafeStreets. It’s based on the idea of safeness; therefore, the goal is to make streets more secure, with less violations, working along with authorities.

All users help each other to know better the street-violation view of the territory.

The system allows both private customer and authority to use the application. They are allowed to sign up entering a username, password and data.

The user is now signed in and can start using the application functions. For the reporting system function, on the main menu it’s possible to click either on “Make a reporting” or “Camera”. The first option allows to make any type of street reporting (accident, speed violation, ecc), while the second is specifically for parking reportings (remember that a picture can be sent only for parkings). The user can now insert all the information about the violation he wants to report. Mandatory information are type of violation, geographical position, which can be either typed in as a address or chosen on the map, and time (license plate not mandatory).

Before the upload of the reporting, the system asks if the reporting wants to remain anonymous, this means that, when the reporting is selected on the map, the username of the user that made it will not be shown.

Now the user can finish the reporting, unloading it on the application. SafeStreets will store the reporting and update the map.

* + 1. **Map management**

On the map are shown colored areas, based on the frequency in which the violations occur. By default, the map of all types of violations that occurred during last month is shown. The user can select a specific type of violation and interval of time in which he is interested in watching the map, for example, parking violations that occurred today.

The user can click on a specific colored area, the map will zoom in and the spots of single violations is now on the map. The user can select a single violation, the information about it will pop up (time, position, license plate and picture if present, username if not anonymous).

* + 1. **Authorities management**

An authority can sign in inserting its authority ID and email address. Authorities, in addition of all the base functions that a private user has, can also send or receive information to/from SafeStreets.

* 1. User characteristics
* User
* License plate algorithm
* Authorities
* Violation
* Reporting
  1. Assumptions, dependencies and constraints

1. Specific Requirements
   1. External interface requirements
      1. User interfaces

The following mockups represent a basic idea of what the mobile app will look like in the first release.

Immagine che contiene monitor, sedendo, schermo, telefono

Descrizione generata automaticamente Immagine che contiene monitor, telefono, schermo, cellulare

Descrizione generata automaticamente

Figure 5 - Login Figure 6 - Menu

Immagine che contiene monitor, telefono, elettronico, cellulare

Descrizione generata automaticamente Immagine che contiene monitor, telefono, cellulare, elettronico

Descrizione generata automaticamente

Figure 7 - Map Figure 8 – Map specifics

Immagine che contiene monitor, automobile, camion, strada

Descrizione generata automaticamente Immagine che contiene monitor, telefono, cellulare, schermo

Descrizione generata automaticamente

Figure 9 – Reporting info Figure 10 – Violation Reporting

Immagine che contiene monitor, telefono, cellulare, schermo

Descrizione generata automaticamente

Figure 11 – Parking violation reporting

* + 1. Hardware interfaces
    2. Software interfaces
  1. Functional Requirements
  2. Performance Requirements
  3. Design constraints
  4. Software system attributes
     1. Reliability
     2. Availability
     3. Security
     4. Maintainability
     5. Compatibility

1. Formal analysis using Alloy
2. Effort spent