RASD document

Version 0.2 – 14/10/2019

Giulio A. Abbo – 10538950

Gianmarco Accordi – 10587213

Massimiliano Bonetti – 10560496

# Introduction

## Purpose

These are the goals of the SafeStreets system:

- G1: The System accepts notifications about the violations.

- G2: The System gives information about the violations to the Municipality.

- G3: The System suggests possible interventions to the Municipality.

- G4: Allow the Municipality to retrieve submitted violations.

- G5: The System gives statistics to the User about the violations.

- G6: The System gives statistics to the Municipality about the violations.

- G7: The System can retrieve the tickets issued by the Municipality

## Scope

Analysis of the world and the shared phenomena.

## Definitions, acronyms, abbreviations

1.c.1 Definitions

* The System: it refers to the software to be

1.c.2 Acronyms

* API = Application Programming Interface
* GPS = Global Positioning System
* UI = User Interface
* S2B = Software To Be

1.c.3 Abbreviations

* Gn = nth goal
* Dn = nth domain assumption
* Rn = nth requirement

## Revision history

Text

## Reference documents

* Specification document: “Mandatory Project Assignment AY 2019-2020”

## Document structure

The RASD document comprises six chapters:

**1. Introduction**:

**2. Overall Description:**

**3. Specific Requirements:**

**4. Formal Analysis using Alloy:** it contains the model, described using the formal language Alloy, of the most critical parts of the system and of the environment. This chapter contains also some worlds obtained from the model and the checks of some assertions.

**5. Effort Spent:** It contains the number of hours used by each member of the group to work on each part of the document.

**6. Reference Documents:** It contains the documents used to write this document.

# Overall description

## Product perspective

Details on the shared phenomena and a domain model (class and states diagrams).

The SafeStreets system is designed as a Progressive Web App, <https://developers.google.com/web/progressive-web-apps>

The world in which the SafeStreets system will work is modelled as follows.

Traffic violation

Municipality makes

Traffic tickets

User finds violation

Violation memorization

User memorization

Plate recognition

Data analysis

User notifies a violation

Municipality accesses to violations

Suggest intervention

**WORLD**

**MACHINE**

Violation’s position sending

User requests statistics

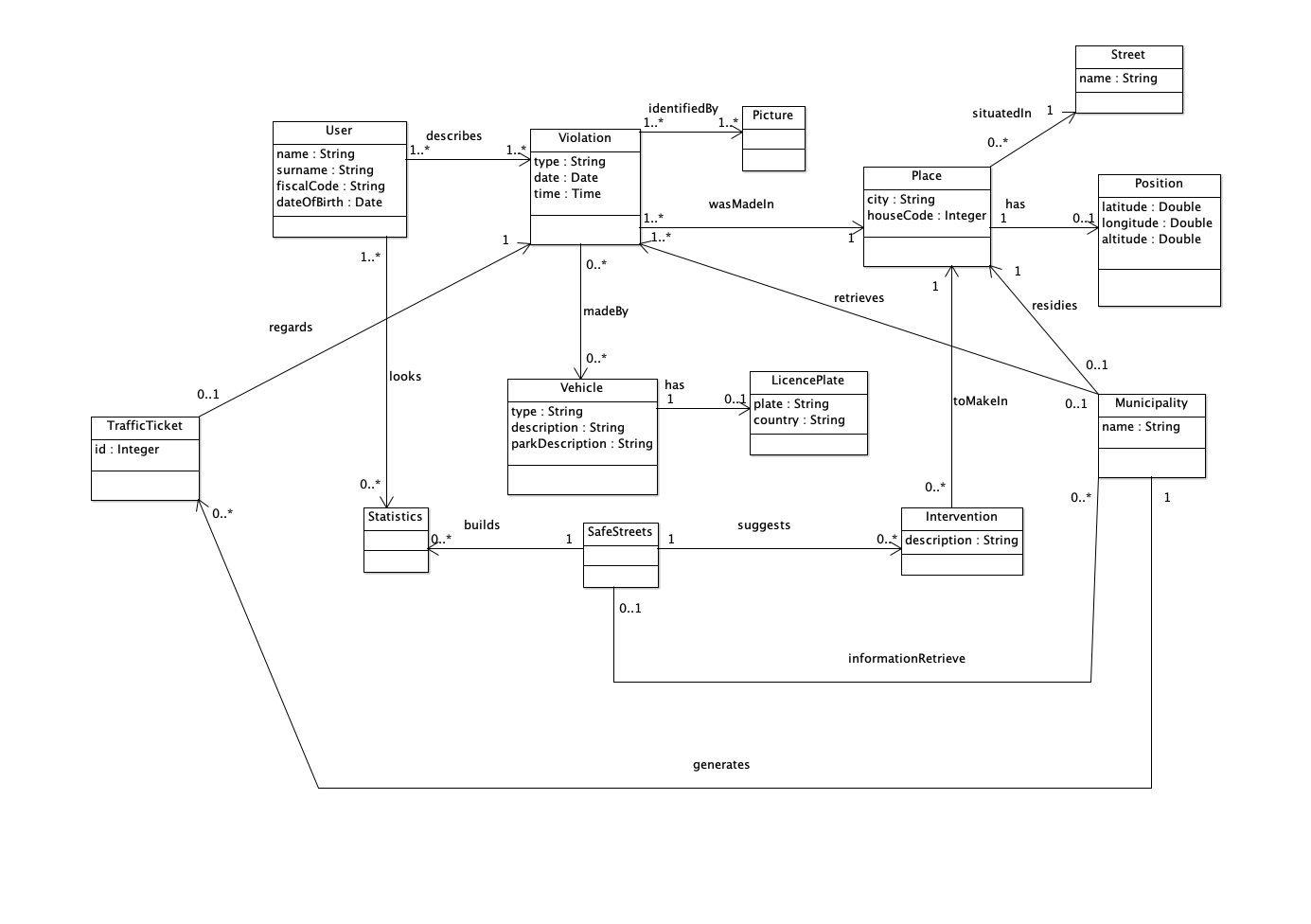
Access to traffic tickets

The notify violations is a shared phenomena that is triggered by the User (that has to be registered in the system) or it can be retrieved by the services offered by the municipality , the violation includes the date, the time, one or more pictures with the main picture contains the License Plate that will be recognized by the system or inserted by the user, the position that can insert by the user or retrieved by the Location system of the device, the type of the violation and the Users that has notify the violation. The violation accessed by the service offered by the Municipality are verified, while the violations notify by the users need to be verified.

The SafeStreet system has also to perform Data Analysis on the streets and vehicle by highlighting the highest frequency violations, that can be useful for the User and the Municipality, that will have different level of granularity: so the user will see the aggregate Data( for example he can see the street with the highest violations), while the Municipality can see also who committed the violations.

Another Shared Phenomena is triggered by the Data Analysis that by looking at the violation frequency can suggest the road that need some intervention.

The next picture contains the class diagram representing the conceptual model of the application domain.



## Product functions

The requirements of the SafeStreet system are:

- R1: The notifications about the violations are correctly stored.

- R2: User must be registered to notify a violation.

- R3: The user can view only the most unsafe areas, the streets with the highest number of violations and the effectiveness of the system.

- R4: The system accepts the violations only of its competence area. ???

- R5: Violations registered by the Municipality can be retrieved by the system.

- R6: The system must avoid the manipulation of the violations.

- R7: The System allows the Municipality to query the violations.

- R8: Only the Municipality can access the violations.

- R9: The User can only access aggregated data. ???

- R10: The system accepts notification from the user also if the municipality is not registered.

- R15: The system must calculate the most unsafe areas.

- R16: The system must calculate the streets with the highest number of violations.

- R17: The system must calculate the effectiveness of the service.

- R18: The system must calculate the vehicles that commit the most violations.

- R19: The municipality can view all the statistics calculated by the system and all the violations.

- R20: Every month the system notifies the municipality with the most unsafe streets and the vehicles that have committed the most violations.

## User characteristics

Anything that is relevant to clarify their needs.

## Assumptions, dependencies and constraints

The domain assumptions of the SafeStreet system are:

- D1: Trust the notification made by the user. (?)

- D2: The plates are correctly recognized.

- D3: The positions are correctly retrieved.

- D4: The Identity card is correctly verified. (?)

- D7: The chain of custody of the violation is never broken if and only if the information about the violation is never altered.

- D9: The Municipality possesses only real violations.

# Specific requirements

## External interface requirements

### User interfaces

Text

### Hardware interfaces

It requires the camera of the smartphone

### Software interfaces

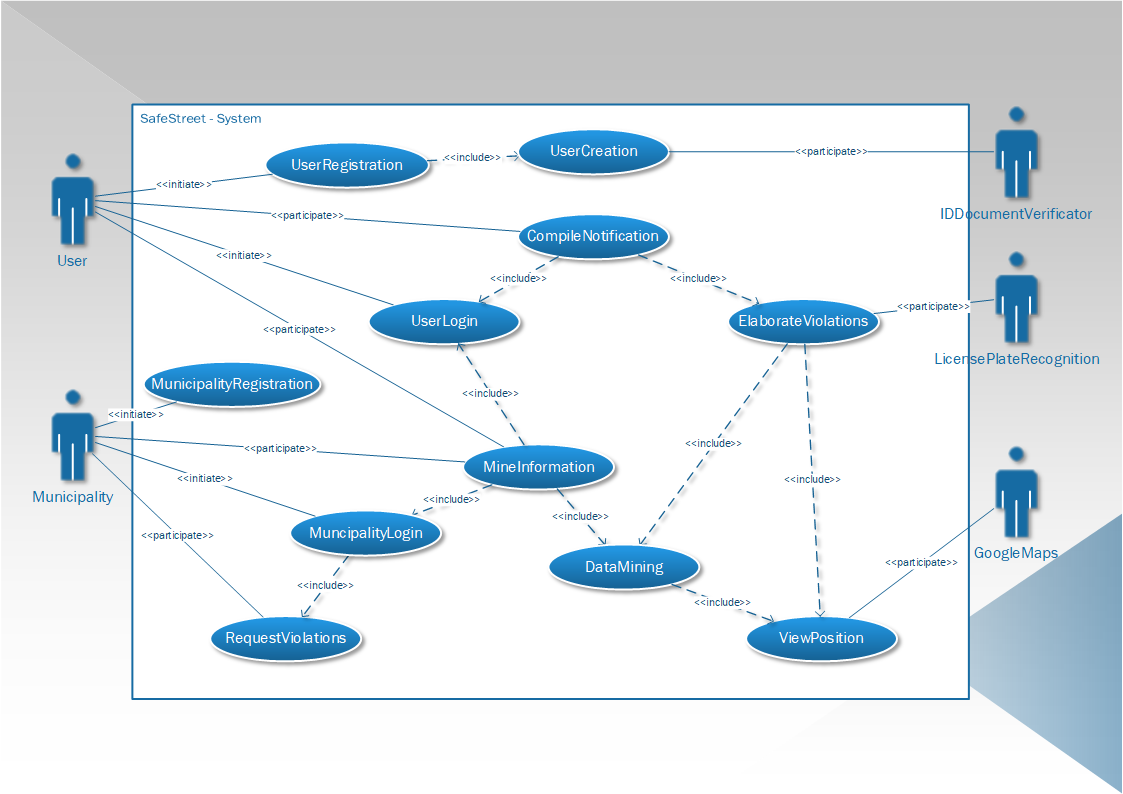
Google Maps  
Traffic Plate recognition  
(Identity card recognition?)

### Communication interfaces

Text

## Functional requirements

**Use Case**

****

**introduction**

**User Registration**

**Actors**: User, ID Document Verificator.  
**Entry conditions**: The User wants to registrate himself inside the service.  
**Flow of events**:

1. The Users sends the request of Registration
2. The Users starts to fill the registration forms
   1. A picture of the User is registered
   2. The User inserts his generality
   3. The User provides also his email and the password he will use
   4. The document of the user is inserted
3. The user then sends the form compiled and the system take care of it
4. The system validates the user identity and verify if another user already exists with the same generality
5. The system contacts a Document Verificator in order to find out if the document provided by the User is valid.

**Exit conditions:** The User has been correctly registered.  
**Exceptions:** Errors occurs if the Identity Document provided by the user cannot be validated by the Document Verificator, or if there exists another User with the same generality.

**User Login**

**Actors**: User  
**Entry conditions**: The User wants to access his account   
**Flow of events**:

1. The User fills the form with his generality and tries to sign in
2. The System receives this request and search the requested account
3. The System verify if the password is correct for the requested account
4. The System responds to the Client and give it the possibility of managing his account or to send some violations notifications

**Exit conditions:** The User has correctly sign in in his account  
**Exceptions:** If the User provide an email that doesn’t appears inside the User List, if the provided password is wrong, also if the User is already logged in.

**Violation Notification**

**Actors**: User, Google Maps, License Plate Recognition  
**Entry conditions**: The User wants to notify a Violation   
**Flow of events**:

1. The User starts to fill a form for the Violations notification
2. The System asks the User for a picture of the violation with the license plate of the vehicle
3. The User can select a picture from the gallery of make a shoot from his camera
4. The System receives the picture inserted by the User and ask the User to fill some required options
   1. The System asks the User to input the Position of the Violations
   2. The User can use the position retrieved through his smartphone or he can insert the position manually
   3. The System then ask the User to select the violation type
   4. The User insert the violation type
   5. Then the System ask the user to fill some options that are not mandatory
   6. The User responds with the non-mandatory options
5. The System than elaborates the violation
6. The System sends the pictures to the Licence Plate recognition services
7. The System elaborate the acquired position through the Google Maps services
8. The System store the violation, and it informs the User about the success of the operation

**Exit conditions:** Violation has been correctly registered  
**Exceptions:** The License Plate Recognition services cannot understand the License Plate ID, the Google Maps services cannot retrieve the position,

In this section we indicate for each goal the requirements and the domain assumption to ensure the correlated goal:

* **G1: The System accepts notifications about the violations.**
  + R1: The notifications about the violations are correctly stored.
  + R6: The system must avoid the manipulation of the violations.
  + R10: The system accepts notification from the user also if the municipality is not registered.
  + D2: The plates are correctly recognized.
  + D3: The positions are correctly retrieved.
  + D7: The chain of custody of the violation is never broken if and only if the information about the violation is never altered.
* **G2: The System gives information about the violations to the Municipality.**
  + R1: The notifications about the violations are correctly stored.
  + R6: The system must avoid the manipulation of the violations.
  + R7: The System allows the Municipality to query the violations.
  + R8: Only the Municipality can access the violations.
  + D7: The chain of custody of the violation is never broken if and only if the information about the violation is never altered.
* **G3: The System suggests possible interventions to the Municipality.**
  + R1: The notifications about the violations are correctly stored.
  + R6: The system must avoid the manipulation of the violations.
  + R15: The system must calculate the most unsafe areas.
  + R16: The system must calculate the streets with the highest number of violations.
  + R17: The system must calculate the effectiveness of the service.
  + R18: The system must calculate the vehicles that commit the most violations.
  + R19: The municipality can view all the statistics calculated by the system and all the violations.
  + D7: The chain of custody of the violation is never broken if and only if the information about the violation is never altered.
* **G4: Allow the Municipality to retrieve submitted violations.**
  + R1: The notifications about the violations are correctly stored.
  + R6: The system must avoid the manipulation of the violations.
  + R19: The municipality can view all the statistics calculated by the system and all the violations.
  + R20: Every month the system notifies the municipality with the most unsafe streets and the vehicles that have committed the most violations.
  + D7: The chain of custody of the violation is never broken if and only if the information about the violation is never altered.
* **G5: The System gives statistics to the User about the violations.**
  + R1: The notifications about the violations are correctly stored.
  + R3: The user can view only the most unsafe areas, the streets with the highest number of violations and the effectiveness of the system.
  + R6: The system must avoid the manipulation of the violations.
  + R15: The system must calculate the most unsafe areas.
  + R16: The system must calculate the streets with the highest number of violations.
  + R17: The system must calculate the effectiveness of the service.
  + D7: The chain of custody of the violation is never broken if and only if the information about the violation is never altered.
* **G6: The System gives statistics to the Municipality about the violations.**
  + R1: The notifications about the violations are correctly stored.
  + R6: The system must avoid the manipulation of the violations.
  + R15: The system must calculate the most unsafe areas.
  + R16: The system must calculate the streets with the highest number of violations.
  + R17: The system must calculate the effectiveness of the service.
  + R18: The system must calculate the vehicles that commit the most violations.
  + R19: The municipality can view all the statistics calculated by the system and all the violations.
  + D7: The chain of custody of the violation is never broken if and only if the information about the violation is never altered.
* **G7: The System can retrieve the tickets issued by the Municipality** 
  + R5: Violations registered by the Municipality can be retrieved by the system.
  + R6: The system must avoid the manipulation of the violations.
  + D7: The chain of custody of the violation is never broken if and only if the information about the violation is never altered.
  + D9: The Municipality possesses only real violations.

For User we intend the person who has been registered and has made the login.  
For Municipality we intend that Municipality which has been registered and has made the login.

## Performance requirements

The software should be used without waiting times, apart from when the application is uploading the violations or when is downloading the statistics from the server, whose time depends on the internet connection.

## Design constraints

### Standard compliance

Don’t broke the chain of custody.

### Hardware limitations

Text

### Any other constraint

Text

## Software system attributes

### Reliability

Text

### Availability

Text

### Security

Text

### Maintainability

Text

### Portability

Text

# Formal analysis using ALLOY

This section should include a brief presentation of the main objectives driving the formal modelling 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 world obtained by running it, and/or the results of the checks performed on meaningful assertions.

Proof that the chain of custody is never altered.

# Effort spent

Text

# References

Text