POLITECNICO DI MILANO

SOFTWARE ENGINEERING II

SafeStreets - Requirements Analysis and Specification Document

Version 1.0

Authors
Iacopo MARRI
Manuel SALAMINO
Steven Alexander SALAZAR MOLINA

October 24, 2019

Contents

1 Introduction

The Requirement Analysis and Specification Document (RASD) aims to focus on the tasks needed to develop and implement an application, taking account of the requirements of the involved stakeholders and, analyzing and documenting also the application requirements.
Then, in the second part of the document, there is a more formal definition of the requirements with the use of the Alloy language.
In general this document is meant for developers tasked with the implementation of the System and also for all the other entities involved in validation and managing of the project.

1.1 Purpose

SafeStreets is an application that was created with the intention of monitoring the compliance with traffic regulations. Its goal is to allow Users to notify traffic violations, collect those data about them, elaborate into information, and then provide them to both Users or authorities who need, for aiming different scopes.

The application provides Users a way to send data about a violation (most common are parking violations ones, e.g. vehicles parked in reserved places), like the kind of violation, pictures of the involved vehicles, and the date and the position in which the violation occurred. The User can also specify the license plate of the vehicle and the name of the street, but in case he/she doesn't, the application is equipped with algorithms capable to obtain those and other metadata by pictures and position.

SafeStreets stores all these data, received ones and computed ones, and allows Users and competent authorities to access them, in order to gather useful information about traffic and violations, e.g. areas with the higher number of violations. This could, for example, help municipality to identify areas in which they must watch over.

Moreover, the System is also able to combine its own data, with data provided by municipality authorities, in case they offer a service that provides them. Basing on the information obtained in this way, SafeStreets will give suggestions about how to improve urban mobility situation, or about how to prevent some kind of violations from being committed (e.g. putting some cameras in vulnerable areas, add some kind of barrier and so on).

For last, authorities are allowed to use information provided by the application and generate traffic tickets. This leads the application to having to implement a method for ensuring the integrity of the incriminating data,

from when the data is generated, to when it is delivered to the application. authorities have to be sure, for example, that they're not giving a fine to a vehicle just because of a picture alteration has been carried out by a notifying User, so altered information must be discarded. SafeStreets will also use statistics on those "rotten" data with the aim to do measure on number of bad Users, or to monitor the reliability and effectiveness of the application it self.

Here below are listed out the goals we did talk about:

- [G.1] Allows the User to access the functionalities of the application from different locations and devices.

- [G.2] Allows the User to notify about traffic violations.

- [G.3] Allows the User to send pictures and kind of the violation, and other information like the license plates of involved vehicles.

- [G.4] Must attach other metadata to the data sent by the User.

- [G.5] Must be able to obtain data like the license plates, or the name of the street involved in the violation, by the pictures and the position sent by the User.

- [G.6] Must store all of these information in a secure way.

- [G.7] Allows both normal Users and authority Users to access its data, and gather relevant information about streets and violations (e.g. streets with more violations).

- [G.8] Must cross its data with the municipality ones, if they provide an interface for allowing other users to access them.

- [G.9] Must provide suggestions to improve traffic safety.

- [G.10] Must ensure that if any corrupted information is provided by Users, it get discarded.

- [G.11] Must use altered data to provide statistic analysis.

- [G.12] Allows Users with authority permissions to generate traffic tickets from its information.

1.2 Scope

According to the World and Machine paradigm, introduced by M. Jackson and P. Zave in 1995. We can idenfity the Machine as the System to be developed and the environment in which SafeStreets will be used as the World. The separation between these two concepts allow us to classify the entire phenomena in three different types.

World phenomena, events that take place in the real world and that the machine cannot observe.

- The driver has an accident and leaves the car in an inappropriate place.

- A malicious user reports a fake traffic violation.

- A user has an old mobile phone with a low quality camera.

- Movement of a user from a position to another one before sending the picture.

- Unexpected connection losses before receiving a picture.

Machine phenomena, events that take place inside the *System* and cannot be observed by the real world.

- Encryption of sensitive data.

- All operations performed to store/retrieve collected data.

- The System retrieves information about unsafe areas from municipalities' services.

- The System manages multiple reports of the same traffic violation.

Shared phenomena:

Controlled by the world and observed by the machine.

- A guest can sign up to the application or log in if is already registered.

- The User can send report traffic violations at any time.

- The Municipality offers up-to-date information about accidents on the territory.

Controlled by the machine and observed by the World.

- The System sends traffic violations to authorities.

4

- The System allows users to view own reports.

- The System shows inferred safe/unsafe areas.

- The Municipality gets suggestions generated by the System.

- The System allows authorities to generate traffic tickets

- The System notifies authorities about adulterated pictures.

- The User can view statistics built by the System.

1.3   Definitions, Acronyms, Abbreviations

1.3.1   Definitions

- *User*, a person which use the application by sending notifications of traffic violations

1.3.2   Acronyms

1.3.3   Abbreviations

- [**G.i**] : i-th goal

- [**R.i**]: i-th functional requirement

1.4   Revision history

1.5   Reference Documents

- Alloy Documentation

- Project assignment specifications

1.6   Document Structure

Section 1 introduces the problem and describes the purpose of the application SafeStreets. Furthermore, describes the scope in which the application is defined by stating the goals and a brief description of phenomena.

Section 2 presents the overall description of the project. *Product Perspective* give more details about the boundaries of the system and world, machine and shared phenomena, while in *Product Functions* are described the main functions of the system and in *User Characteristics* the main actors. At last are defined the domain assumption on which the system relies on.

Section 3 contains all the specific requirements needed to satisfy each goal.

Furthermore, are highlighted the major functions and interactions between the actors and the system using use cases and sequence diagrams.

Section 4 shows the alloy model and discuss its purpose.

Section 5 explain the effort spent by each group member to accomplish this project.

2 Overall Description

2.1 Product perspective

Here we discuss in details all the shared phenomena outlined in Section 1.3 and we also provide a domain model through class and state diagrams.

Shared Phenomena, controlled by the world and observed by the machine.

- Register/login: a normal person or a municipality worker can register to the application through two different intuitive forms; SafeStreets collects all the information inserted by the User (normal person or municipality worker), creates an account and verify the validity of the information given through a email-confirmation message. Once the User confirms its account he/she can proceed to login.

- Report of a traffic violation: the User is able to report traffic violations at any time, he/she can open the application, select *take a picture* and then send the report including the picture, the type of violation and the name of the street where the violation occurred.

- The Municipality offers up-to-date information about accidents: in the Municipality web platform there is a service (daily updated) that indicates streets in which accidents occurred. In this way the System can observe safe/unsafe areas.

Shared Phenomena, controlled by the machine and observed by the world.

- Forwarding of traffic violations: every time a user sends a traffic violation, the System forwards it to the authorities. The message sent by the System contains the photo of the violation, the type of violation, the street in which it took place and the area in which the photo was taken (since there may be more than one street with that name).

- Visualization of own reports: the System makes it possible for the users to view the own reports by performing a query to the database and displaying them in a ListView in the application.

- Visualization safe/unsafe areas: the System through the service offered by the municipality is able to get a list of areas in which accidents took place, thus the System can show unsafe and safe areas to the User.

- Suggest possible interventions: the System, by using a list of known and effective solutions for some common violations, is able to suggest possible interventions in order to reduce the number of violations and make unsafe areas safer.

- Generate traffic tickets: the municipality, by observing the reports provided by Safe4Streets, is able to generate traffic tickets by using a functionality of the application.

- Notify about modified photos: if the User modifies a photo in order to send malicious reports, the system mark the User as a malicious User and sends a message about this event to authorities (that already received reports created by the User).

- Generate statistics: the System is able to generate graphs that illustrate statistics, like the most egregious offenders, the effectiveness of the application itself, etc.

2.2 Product functions

SafeStreets

SafeStreets was born with the intention of improving urban traffic mobility, by collecting notifies made by Users about many kinds of traffic violations. Then these information can be consulted by Users themselves, or from a Third Part, the traffic competent authorities, or the municipality, which can create tickets based on these information, on which integrity it is necessary to pay particular attention. Indeed, *SafeStreets* provides useful suggestions built upon the crossing of municipality's and *SafeStreets*'s data. Let's see more in the details the functions mentioned above:

- **Notifying violations**

  The Users who is witness of a rule violation, and wants to notify it through *SafeStreets*, has to be registered to the application; then he/she can fill up the notification form provided by the application, with a picture or a video of the violation occurring, the kind of violation, the kind of vehicle, the position, (or the address, which can be anyway gathered from GPS), and the license plate of the indeed vehicles.
  Not all of these data are mandatory for the User to send: *SafeStreets* uses an algorithm to catch the license plates from the picture, can obtain the address by the GPS position, and it implements a deep learning algorithm to recognize the vehicles it self. *SafeStreets* is going to attach these computed metadata to the notification, in case of missing.

- **Accessing violations data**

Registered Users (quelli non registrati possono?) and authorities can access the information stored by the application.

SafeStreets provides an easy-use interface that allows to carry out a search for violations based upon different criteria: the *kind* of the violation, the *place* in which happens, the *time*, and the *kind of vehicle* involved. A User can, for example, to search for a place, and than see at which time slot of the day, more violations are committed in that place, or in which place or at which time slot, a precise type of violation occurs the most.

By the way, due to privacy purposes, license plates related to violations are not visible by a normal User, neither a User can search some information about a precise vehicle with a precise license place. Only authorities can access those "more sensible" data,

(Da qui alla fine dell'item andrà spostato tutto nella sezione dei functions per le authorities)

and in order to build this authorization diversification, a different type of registration will be reserved for authorities Users. (Da decidere come)

- **SafeStreets' suggestion mechanism**

  If the local municipality provides a way to access their own urban traffic and violations data, *SafeStreets* can cross them, to enlarge the amount of information accessible, and exploit them to formulate suggestions in order to increase the functionality of the urban infrastructure, reduce the violations, improve road conditions.

  If we imagine a situation in which *SafeStreets* knows about huge amount of bike lane invasion violations, and municipality knows that in that zone, many people use bikes for moving, *SafeStreets* can cross those information and provide the suggestion to build a separation line between the car road, and the bike lane, because it should be a good investment, knowing the fact that not only there are lots of violations, but also that those could be very dangerous, given the number of people using bike in that zone.

- **vado a cena e finisco ;);););););)(;(;(;(;(;(;(;(;**

2.3 User characteristics

2.4 Assumptions, dependencies and constraints

3   Specific Requirement

3.1   External Interface Requirements

3.1.1   User Interfaces

3.1.2   Hardware Interfaces

3.1.3   Software Interfaces

3.1.4   Communication Interfaces

3.2   Functional Requirements

3.3   Performance Requirements

3.4   Design Constraints

3.4.1   Standards compliance

3.4.2   Hardware limitations

3.4.3   Any other constraint

3.5   Software System Attributes

3.5.1   Reliability

3.5.2   Availability

3.5.3   Security

3.5.4   Maintainability

3.5.5   Portability

4 Formal Analysis using Alloy

5 Effort Spent

6 References