POLITECNICO
MILANO 1863

SafeStreets

Software Engineering 2 Project

Salvatore Fadda, Adriano Mundo, Francesco Rota

A.Y. 2019/2020
Version 1.0

November 10, 2019

Contents

1 Introduction

This document represents the Requirements Analysis and Specification Document (RASD). It aims at providing an overview of the project *SafeStreets*. It illustrates the purpose of the project, starting from its goals and how these can be reached ensuring certain non functional requirements, functional requirements and constraints. This document is intended for all the people involved in the project life-cycle, from planning and estimation to development and validation.

1.1 Purpose

SafeStreets is a crowd-sourced application that aims at keeping safe the city's streets. The application goal is to allow *Users* to notify the *Municipality* when a violation occur on the streets under its jurisdiction. The *User* can notice and notify the violation by sending a photo of the violation including date, time and position. These violations are for the majority parking violation, such as double parking or vehicle parked in the middle of bike lanes.

SafesStreets, once the *User* has notified the violation, stores all the data, completing them with all the necessary metadata. In order to be sure that the violation is correctly elaborated, the application uses a plate recognition algorithm. The *User* is notified if something goes wrong during the whole process, so an alternative solution can be found. All the data stored by *SafeStreets* are provided by the *Users* or they can be retrieved directly from the device, like the position from the GPS system.
All the data collected by *SafeStreets* can be mined by *Users* and the *Municipality*, so they can be provided with statistics built from this data. The application can show different statistics based on different level of visibility, so the *Municipality* can access to some information that the *Users* can not access, and viceversa. This is the purpose of the *Basic Service*.

The application has also two specific *Advanced Functions*. In the **AF1**, the application *SafeStreets* can cross the information about the road accident, thanks to a service offered by the *Municipality*, with the data stored and retrieved from the *User*. So *Municipality* elaborates them in order to identify potentially unsafe areas and suggest possibile interventions to solve founded issues. The suggestions can be various and depend on each case.

In the **AF2**, *SafeStreets* allows the *Municipality* to generate traffic tickets directly from the application data, derived from the user notifications'. Additionally, starting from the application data about issued tickets, *SafeStreets* can build, looking for trends in the data, statistics and provide insights to the *Municipality*. This can help *Municipality* to improve the process of issued tickets and understand the effectiveness of *SafeStreets*, finding some information useful to improve their services.

From the description above about the purpose of *SafeStreets*, we can summarise those goals

- **[G1]** Allowing *Users* to report to the system when a traffic violation occur.
- **[G2]** Allowing *Users* to enter data/information about the violation.
- **[G3]** Providing both *Users* and *Municipality* with statistics built from notifications data.

  With regards to *Advanced Function 1*, we identify:

- **[G4]** Identifying potentially unsafe areas and making suggestions to address those issues.

  With regards to *Advanced Function 2*, we identify:

- **[G5]** Allowing *Municipality* to generate tickets based on the users notifications.
- **[G6]** Providing statistics built on data from issued tickets to the *Municipality*.

1.2 Scope

In this section we will distinguish between the *Machine*, that's the **S2B** and the *World*, that's the portion of the real world affected by the *Machine*. This separation, according to the *World and Machine* paradigm by Jackson and Zave, leads to a classification of the phenomena in three different types, depending on where they occur.

1.2.1 World Phenomena

World phenomena are events that take place in the real world and do not have a direct impact on the sytem.

- A generic traffic violation occurs on the streets under the *Municipality's* jurisdiction.
- An accident occurs on the streets under the *Municipality's* jurisdiction.
- A *User* notices the violation and takes action.
- A ticket is generated by the *Municipality*.
- Data from all the tickets generated by the *Municipality* are stored by the *System*.
- An intervention is made by the *Municipality* to address possible issues on the streets under its jurisdiction.

1.2.2 Shared Phenomena

Shared phenomena are world phenomena that are shared with the Machine. These are further divided in two categories.

Controlled by the world and observed by the machine

- A *Guest* signs up on the system giving all the personal data needed and/or signs in with his credentials.
- A *User* sends a violation notification to the system, including type of violation, position, date, time and picture.
- *Municipality* transmits data about the accidents occurring on the streets to the *System*.
- *Municipality* evaluates a violation notification coming from the *System*.
- *Municipality* elaborates a suggestion for an intervention coming from the *System*.
- *Municipality* transfers data about tickets generated to the *System*.

Controlled by the machine and observed by the world

- The *System* notifies the *User* that the plate recognition tool was not successful, in order for the user to take a better picture.
- The *System* transmits a violation notification to the *Municipality*.
- The *System* provides suggestions on possible interventions to address issues to the *Municipality*.
- The *System* shows statistics mined from violation notifications data to *Users* and/or the *Municipality*.

1.2.3 Machine Phenomena

Machine phenomena are events that entirely take place inside the System and cannot be observed in the real world.

- The plate recognition algorithm is ran on the picture of the violation.
- The complete set of data are stored and can be retrieved by the *System's* DBMS.

1.3 Definitions, Acronyms, Abbreviations

1.3.1 Definitions

- **Guest:**
- **Authorities:**
- **User:**
- **Municipality:** Entity composed of both men and the information system that has authority on the streets, where it has the responsibility to enforce the rules and to guarantee safety.
- **Violation:** the action of violating traffic laws.
- **Ticket:** administrative sanction established by law for a violation.
- **Plate recognition Algorithm:** algorithm that automatic recognise vehicle's' plate by the images sent from users to report a violation or an accident.
- **Notification Data:** information that is provided by the user when he reports a violation. This includes picture, license plate, date, time, position.
- **Ticket Data:** information that is provided by the municipality when it adds a new ticket in the system. This includes violation type, license plate, date, time, position.
- **Accident Data:** information that usually is provided by the municipality when it reports an accident. This can includes accident type, picture, multiple license plate, date, time, position.
- **Intervention:** action taken by the municipality to prevent further issues in the city traffic.
- **Notification:** message sent by the user to advise the system about a violation.

1.3.2 Acronyms

- **GPS:** Global Positioning System
- **API:** Application Programming Interface
- **ID:** Identifier
- **RASD:** Requirements Analysis and Specification Document
- **DBMS:** Data Bases Management System
- **GDPR:** General Data Protection Regulation

1.3.3 Abbreviations

- **[Gn]:** n-th goal
- **[Rn]:** n-th functional requirement
- **[Dn]:** n-th domain assumption
- **AF1:** Advanced Function One
- **AF2:** Advanced Function Two
- **SP1:** Shared Phenomena controlled by the World and observed by the Machine
- **SP2:** Shared Phenomena controlled by the Machine and observed by the World
- **WP:** World Phenomena
- **MP:** Machine Phenomena

1.4 Revision History

Version	Date	Changes
1.0	04/11/2019	First Draft
1.1	10/11/2019	First Release

Table 1: Revision History

1.5 Reference Documents

- Mandatory Project Assignment
- Alloy Official Documentation: http://alloy.lcs.mit.edu/alloy/documentation.html
- ISO/IEC/IEE 29148: System and Software engineering - Life cycle process - Requirements engineering

1.6 Document Structure

The rest of the document is organised as follows:

- **Overall Description** (Section 2): it will be given a general description of the application, with an analysis of the domain focusing on descriptions about the phenomena according to the *World and the Machine* paradigm, and the User Characteristic. It will be provided class diagram and state charts in UML Language and also domain assumptions, dependencies and constraints.

- **Specific Requirements** (Section 3): in this section all the Functional Requirements of the application are explained in details and related to use case scenarios and sequence diagrams clarifying process and interactions between the actors and the *System*. Also, there are descriptions of all the non-functional requirements and external interfaces.
- **Formal Analysis** (Section 4): description and creation of simulation using a formal model, the Alloy specification language in order to address the critical aspects of the *SafeStreets System*.

2 Overall Description

2.1 Product perspective

2.1.1 Class Diagrams

2.2 Product functions

2.3 User characteristics

SafeStreets has two kinds of users. There are simple *Users* of the application and *Authorities*. The first type is usually a citizen or a tourist of the city where the *SafeStreets'* service is activated. The *User* needs to notify authorities of a violation, and in order to do this, he/she has to install the application on his/her smartphone. The second type of user are *Authorities*, they are representative of the *Municipality* that needs to access statistics provided by the service or need to generate tickets **(? da rivedere se vogliamo tenere questa frase, in base a come decidiamo di gestire la generazione del ticket)**

2.4 Assumptions, dependencies and constraints

2.4.1 Constraints

- *Users* are located on the streets under the *Municipality* jurisdiction when they notify the violation.
- *Users* must have a smartphone application to access the *System*.
- The *System* must respect the GDPR regulation.
- The *System* must ask the permission to process personal data about *Users*.
- The *System* must guarantee that the information is never altered during the whole process.

2.4.2 Dependencies

- The *System* needs a DBMS service to retrieve and store data.
- The *System* make use of the GPS provided by the smartphone.
- The *System* uses a map visualisation service.
- The *System* make use of internet connection provided by the smartphone.

2.4.3 Domain Assumptions

- **[D1]**
- **[D2]**
- **[D3]**

- **[D4]**