POLITECNICO
MILANO 1863

Software Engineering 2
SafeStreets: RASD

Version 1.0 - (date)

Authors:

Andrea Falanti

Andrea Huang

Professor:

Prof. Elisabetta Di Nitto

Contents

1 Introduction

1.1 Purpose

SafeStreets is a crowd-sourced application that intends to provide users with the possibility to notify authorities when traffic violations occur, and in particular parking violations. The application allows users to send pictures of violations, including their date, time, and position, to authorities. Examples of violations are vehicles parked in the middle of bike lanes or in places reserved for people with disabilities, double parking, and so on.

SafeStreets stores the violation reports provided by users, that can be made from the official mobile application or from the SafeStreets website, so that users and authorities can visualise and analyse the data received by the system, for example by highlighting the streets (or the areas) with the highest frequency of violations, or the vehicles that commit the most violations. The data can be accessed with different levels of visibility, where most sensitive data can only be mined by authorities.

SafeStreets also provide a service aimed at helping municipalities to identify potentially unsafe areas and suggest possible interventions. To have access to these features, the municipalities need to offer a service for retrieving info about accidents in their territory, so that Safestreets is able to cross a municipality's data with its own stored data and also analyse and suggest possible improvements to the areas with most reports (e.g., add a barrier between the bike lane and the part of the road for motorized vehicles to prevent unsafe parking).

1.2 Scope

World events:

- Traffic violations
- Violation detection
- Street interventions and improvements
- Authorities interventions

Shared events:

- Violation report codification
- Data reading and analysis
- Validated violations notification to authorities
- Intervention suggestion to municipality

Machine events:

- Database queries

- Possible interventions computation
- Licence plate recognition
- Meta-data completion

Goals:

G1. Report actual traffic violations to authorities

G2. Store data about violations and cross it with municipality's ones to identify possible interventions

G3. Provide a synthesis of all the data to public and improve people awareness about the topic

Domain assumptions:

D1. Usernames must be unique

D2. Users report violations when they detect them

D3. Traffic violations must occur regularly

D4. Municipality service about accidents is always available

D5. Municipality accident data is always accurate

D6. Report supervisor always validates the correctness of the reports

D7. The GPS sensor of a user's device has an error of at most 5m from the real position.

D8. The internet connection is always available when the user interacts with the system. (may need to be specified more clearly)

Requirements:

R1. When a traffic violation is reported, the report supervisor verifies it and then notifies the authorities (SE1, DP4, SE3, G1)

R2. Reported traffic violations are stored, integrated with data from the municipality and analysed by an algorithm to find possible interventions to address the problem. Then these suggestions are sent to the municipality to evaluate a possible solution (SE1, DP3, ME2, SE4, G2)

R3. Traffic data is stored and available for public consultation with different levels of visibility, according to user's level of authorization (SE1, SE2, G3)

1.3 Definitions, Acronyms, Abbreviations

1.4 Revision history

1.5 Reference Documents

- Specification document: “SafeStreets Mandatory Project Assignment”
- IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications
- UML diagrams: https://www.uml-diagrams.org/
- Alloy doc: http://alloy.lcs.mit.edu/alloy/documentation/quickguide/seq.html

1.6 Document Structure

2 Overall Description

2.1 Product perspective

Abbott text analysis

Term	Grammar construct	UML Class
SafeStreets	Noun	Main class
Users	Noun	Class
Notify	Verb	Operation
Authorities	Noun	Class
Traffic violations	Noun	Class
Parking violations	Noun	Subclass
Send	Verb	Operation
Pictures	Noun	Object
Date	Noun	Attribute
Time	Noun	Attribute
Position	Noun	Attribute
Vehicles	Noun	Object
Parked	Verb	Association
Bike lanes	Noun	Subclass
Place	Noun	Class
Double parking	Noun	Subclass

2.2 Product functions

2.3 User characteristics

2.4 Assumptions, dependencies and constraints

3 Specific Requirements

3.1 External User 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

3.2.1 User

Scenarios

1. Johnny is a model citizen of Milan and he is really caring about parking violations, because they are really recurrent in his neighborhood. When he see a violation, he open the SafeStreets app on his smartphone and report it with just few clicks. All he needs to do is to take a photo of the violation where the licence plate is clearly visible and confirm if the system detected it correctly or not, if not he insert the licence plate manually in the field that appears after the confirmation. Position is automatically detected from the GPS because he allowed the app to access it, and to send the report to SafeStreets he just need to confirm by clicking the done button.

Use cases

- NotifyViolations:
  - Actors: User.
  - Entry condition: violation detection.
  - Event flow:
    1. The User selects the report tab if not already selected.
    2. The User selects the type of violation.
    3. The User takes and uploads a photo of the violation.
    4. The User confirms that the license plate is correctly recognized by app, if not he/she inserts it manually in the appropriate field.
    5. The User checks whether his/her current position is correctly identified by the GPS system, if not he/she inserts it manually in the appropriate field.
    6. The User confirms the violation report clicking on the "Done" button.

    7. The system stores the information and completes it with suitable meta-data.
    8. The system sends a notification to the User about the success of the operation.
  - Exit condition: the violation report is correctly stored.
  - Exception: the system detects missing information and rejects the report, then asks the User for missing data.
  - Special requirement: TBD.

- AnalyzeData:
  - Actors: User or Authority.
  - Entry condition: the Actor wants to know some information about the reported violations.
  - Event flow:
    1. The Actor selects the "aggregated data" tab.
    2. The Actor selects the topic he/she is interested about.
    3. The Actor may select an appropriate filter for his search.
    4. The Actor selects the data of interest.
    5. The system provides the data to be visualized according to the actor's authorization level.
    6. The Actor visualizes the data.
  - Exit condition: the Actor finishes to analyze the data.

- Login:
  - Actors: User, authority.
  - Entry condition: The actor opens the app.
  - Event flow:
    1. The actor inserts his username.
    2. The actor inserts his password.
    3. The actor press the login button.
  - Exit condition: The actor is authenticated and login is successful.
  - Exception: Username or password are invalid, "invalid credential" message is displayed and the actor needs to insert credential again.

- UserRegistration:
  - Actors: User.
  - Entry condition: The user opens the app and has no valid account.
  - Event flow:
    1. The user inserts the desired username.
    2. The user inserts the desired password.
    3. The user confirms his password.
    4. The user inserts his email.
    5. The user inserts his birthday.
    6. The user agrees to the use of his personal data and of his submitted violation reports for analysis purposes.
  - Exit condition: The registration is successful and the user is redirected to login form.
  - Exception: Email is already in use or confirm password field content diverge from password, in this case the user is alerted with a proper message on screen and asked to correct the data.

3.2.2 Third Party

Scenarios

1. Henry is a member of the local police of the city of Casalpusterlengo, a municipality that have a partnership with SafeStreets. Every month he wants to check the most unsafe areas in the city, so that he could better patrol the city. This could be done by opening the SafeStreets app and selecting "analyze" tab, then selecting the violation density map option and using the previous month as temporal filter.

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

Andrea Falanti:

Document section	Hours
Introduction	4.5
Overall Description	0
Specific Requirements	2.25
Formal Analysis Using Alloy	0
Total	6.75

Andrea Huang:

Document section	Hours
Introduction	2.5
Overall Description	0
Specific Requirements	1.5
Formal Analysis Using Alloy	0
Total	4

6 References