Davide Perugini, Antonio Pipita, Stefano Panzeri

SafeStreets

Requirement Analysis and Specification Document
Software Engineering 2 Project

Deliverable:	RASD
Title:	Requirement Analysis and Verification Document
Authors:	Davide Perugini, Antonio Pipita, Stefano Panzeri
Version:	1.0
Date:	10-November-2019
Download page:	https://github.com/fafrullo2/PanzeriPeruginiPipita
Copyright:	Copyright © 2019, Davide Perugini, Antonio Pipita, Stefano Panzeri – All rights reserved

Contents

List of Figures

List of Tables

1 Introduction

1.1 Purpose

The purpose of this project is to study the requirements and provide a specification about SafeStreets, a crowd-sourced application that permits users to notify authorities about traffic violations.

This document represents the requirements analysis and specification document of SafeStreets, where purposes, goals, requirements and assumptions of the applications will be defined to provide a support for the stakeholders.

SafeStreets allows users to send pictures of traffic violations with every useful metadata about it (date, time, position, type of violation, etc. . .) to authorities.

In addition, the application provides users and authorities with tools to mine the data collected, for example highlighting the streets where parking violations happen frequently, or the most unsafe areas.

SafeStreets also has the possibility to cross its own data with information about accidents coming from the municipality (if the municipality offers this information as an open service) to indentify unsafe areas with more precision and suggest possible interventions.

Finally, if the local police offers a service that takes data and pictures from SafeStreets to generate traffic tickets, the application must ensure the veracity of the information and use data about issued tickets to build statistics (effectiveness of the service, most dangerous vehicles etc. . .).

1.1.1 Goals

- (G1) Allow users to send pictures and informations about traffic violations.
- (G2) Allow users and authorities to mine information collected by the application.
- (G3) The system must recognise (from pictures) and store license plates.
- (G4) The system must be able to retrieve the geographical position where the violation occurred.
- (GA1.1) The system must be able to cross the data collected with information about the accidents coming from the municipality.
- (GA1.2) The system must be able to suggest possible interventions to decrease the risk of violations in unsafe areas.
- (GA2.1) Allow the local police to retrieve data about violations to generate traffic tickets.
- (GA2.2) The system must be able to ensure the veracity of the information sent by the users.
- (GA2.3) The system must track wether the police has taken care of a certain violation.

- (GA2.4) Traffic tickets must be issued to the person that owns the vehicle that committed the violation.

- (GA2.5) The system must be able to build statistics from the information about issued tickets.

1.2 Scope

In a metropolitan city like Milan, one of the biggest problems is the overflowing stream of vehicles in the streets that comes and goes from universities, stations, work etc...

This is the perfect context in which an application like SafeStreets should be developed.

The application is thought for a world in which most of the people always brings along a smart device like a smartphone, able to take photographs and with a stable connection to the internet.

SafeStreets will allow every person of a city to collaborate to make streets safer and help police and municipality to identify areas where violations happen more often.

Data collected by the service, can be later mined by both users and authorities; this can be useful for users, in order to avoid dangerous streets or really messy car parks, and for authorities in order to strengthen controls in the most unsafe areas or to plan possible interventions.

In a world where everyone owns a smart device, it is really easy to modify images so, to avoid fake data sent by the users, SafeStreets will also implement several countermeasures to check the veracity of every piece of information.

1.3 Definitions, acronyms and abbreviations

1.3.1 Definitions

- User: the customer of the application that exploit the service to send pictures and informations about traffic violations
- Municipality: the government of a city; it can mine data from SafeStreets to obtain information about traffic violations and statistics.
- Authorities: comprehend the municipality and the local police.
- Traffic ticket: a fee issued by the local police to people that own a vehicle that committed traffic violation
- Traffic violation: occurs when drivers violate laws that regulate vehicle traffic on streets or parking.

1.3.2 Acronyms

- RASD: Requirement Analysis and Specification Document

1.3.3 Abbreviations

- (Gn): n-Goal
- (G1.n): n-Goal for advanced function 1
- (G2.n): n-Goal for advanced function 2

- (Dn): n-Domain assumption
- (Rn): n-Requirement

1.4 Revision history

1.5 Reference documents

- Specification document: “SafeStreets Mandatory Project Assignment”

1.6 Document structure

- Chapter 1: an introduction to SafeStreets; it describes the purpose and the goals that the application aims to reach. It defines also the scope of the application, that includes the analysis of the world and of the shared phenomena.
- Chapter 2: provides an overall description of the system functionalities. It contains the various charts and diagram describing the domain, the most important requirements, the needs of the users, the various stakeholders and various constraints and assumptions.
- Chapter 3: provide a more specific study about the requirements of the applications describing the various interfaces needed (user, software and hardware), functional requirements with associated use cases and use case/sequence diagrams, performance requirements, design constraints and various software attributes (reliability, availability etc. . .).
- Chapter 4: provides a formal analysis of the application using Alloy. Here will be shown the alloy model of the most critical aspects, various comments to show how the project has been modelled and the world obtained by running the model itself.
- Chapter 5: information about the effort spent by every member of the team on the project.

2 Overall Description

Here you can see how to include an image in your document.

Here is the command to refer to another element (section, figure, table, ...) in the document: *As discussed in Section 1.6 and as shown in Figure 1, ...*. Here is how to introduce a bibliographic citation [1]. Bibliographic references should be included in a `.bib` file.

Table generation is a bit complicated in Latex. You will soon become proficient, but to start you can rely on tools or external services. See for instance this https://www.tablesgenerator.com.

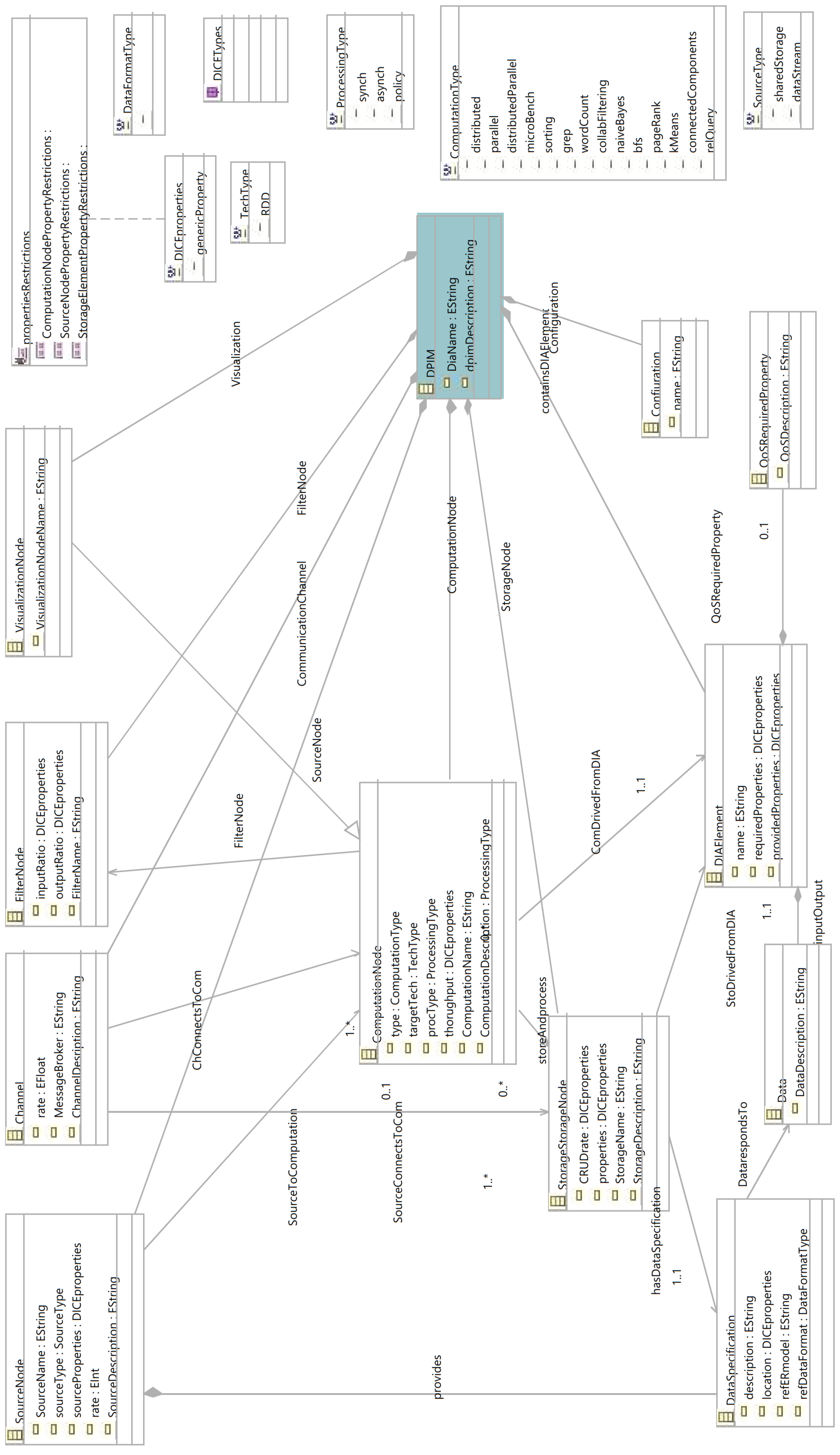

Figure 1: DICE DPIM metamodel.

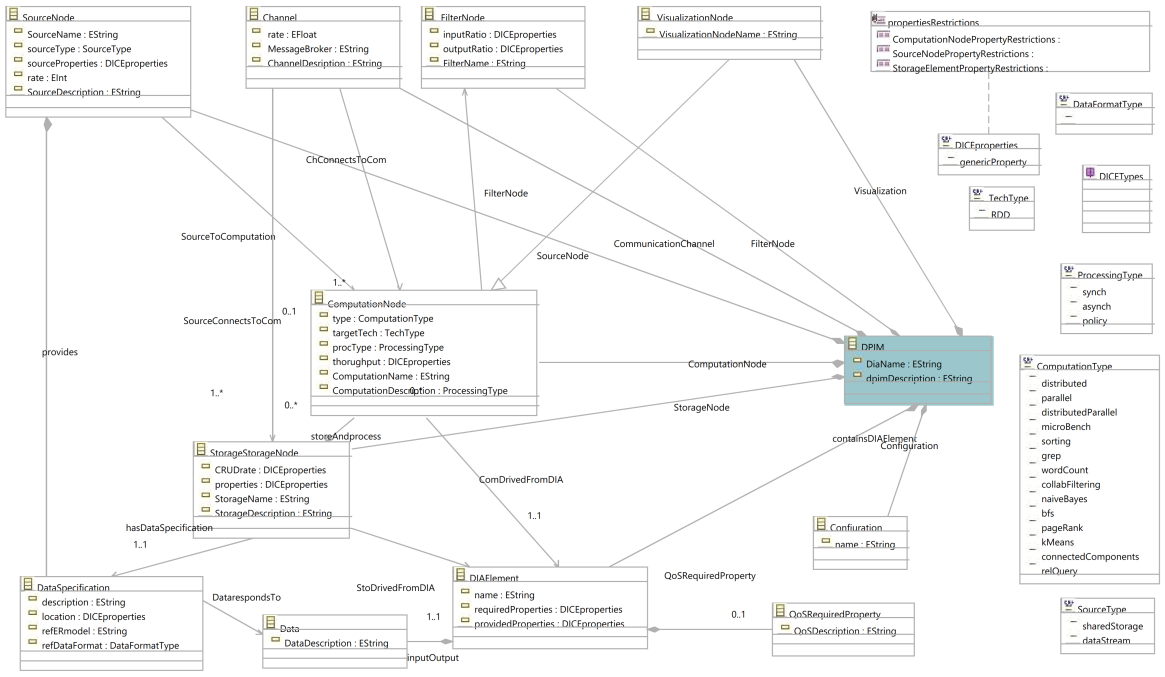

Figure 2: DICE DPIM metamodel in portrait form.

3 Specific Requirements

Organize this section according to the rules defined in the project description.

4 Formal Analysis Using Alloy

This section contains the project analysis done using Alloy

SIGNATURES

```
sig Boolean{}
sig True extends Boolean{}
sig False extends Boolean{}
sig Photo{}
sig CF{}
sig Person{
cf: one CF
}
sig Plate{}
sig Vehicle{
plate: one Plate,
ownedby: one Person
}
sig User{
person: one Person,
areaOfInterest: one Area
}
sig GPScoords{
latitude: one Int,
longitude: one Int
}
sig Intervention{}
sig Area{
segnalationsInside: some Segnalation,
dangerLevel: one Int
interventions: some Intervention
}
sig PositionAndTime{
coords: one GPScoords,
time: one Int
}{time>=0 and time<7}
```

Note: the numbers related to time have been diminished in value for analysis performance reason

```
abstract sig ViolationType{}
sig ExpiredTicket extends ViolationType{}
sig UnauthorizedParking extends ViolationType{}
```

Note: UnauthorizedParking and ExpiredTicket are just two examples of the violations that may occurr

```
sig Segnalation{
maker: one User,
vehicle: one Vehicle,
positionAndTime: one PositionAndTime,
vehicle: one Vehicle,
violationType: one ViolationType,
photo: one Photo,
takenCareOf: one Boolean,
writtenPlate: one Plate }
```

Note: vehicle represents the information retrived from the photo by the system, crossed with the database of car owners; on the other hand writtenPlate is the plate that the user that made the segnalation reports

```
sig Authority{
person: one Person
}
sig MunicipalAuthority{
trackedUsers: set User
trackedArea: set Area
trackedVehicles: set Vehicle
}
sig Policeman extends Authority{}
sig Ticket{
segnalations: one Segnalation,
policeman: one Policeman,
issuedTo: one Person
}
```

FUNCTIONS

```
fun getCoords [s:Segnalation]:GPScoord{
s.positionAndTime.coords
}
fun getTime [s:Segnalation]:Int{
s.positionAndTime.time
}
```

FACTS

```
fact booleanValue{
#True=1 and #False=1 and #Boolean=2 and
(all b:Boolean | b=True or b=False) and
(no b: Boolean | b in True and b in False)
}
fact uniqueFoto {
all p1: Photo | no disj s1, s2 : Segnalation | s1.photo=p1 and s2.photo=p1
}
fact noLonePhoto {
all p1:Photo | p1 in Segnalation.photo }
fact noSameCF {
no disj p1, p2: Person | p1.cf=p2.cf }
fact getCoords {
}
fact noSamePlate {
no disj vei1, vei2: Vehicle | vei1.plate=vei2.plate }
fact noDoubleJob {
no p:Person | p in MunicipalAuthority.person and p in Policeman.person
no disj p1, p2: Policeman| p1.person=p2.person
no disj ma1, ma2: MunicipalAuthority | ma1.person= ma2.person
no disj u1, u2: User| u1.person=u2.person
}
fact cityLimits {
all gps: GPScoord | gps.latitude>0 and gps.longitude>0 and
```

```
gps.latitude<7 and gps.longitude<7
}
fact noDoubeCoordinates {
all c1: GPScoord | no c2: GPScoord | c1 != c2 and
c1.longitude=c2.longitude and c1.latitude= c2.latitude
}
fact areaProperties {
all a: Area| #a.segnalationsInside>=0 and
dangerLevel=#a.segnalationsInside
}
fact noMissmatchingPlates {
all s: Segnalation| s.vehicle.plate=s.writtenPlate
}
fact allPlates {
#Segnalation.writtenPlate=#Vehicle
}
fact violationTypeCardinality {
#ViolationType=2 and #ExpiredTicket=1 and #UnauthorizedParking=1
}
fact allSegnalationsInAnArea {
all s: Segnalation | s in Area.segnalationsInside
}
fact eitherAllTakenCareOrNone {
all s1, s2: Segnalation | (getCoords[s1]=getCoords[s2] and
getTime[s1]=getTime[s2] and s1.writtenPlate=s2.writtenPlate
and s1.violationType=s2.violationType) implies
(s1.takenCareOf=s2.takenCareOf)
}
fact sameVehicle {
all disj s1, s2 : Segnalation | all t: Ticket |
(s1 in t.segnalations and s2 in t.segnalations) implies s1.vehicle=s2.vehicle
}
fact takenCareOfRule {
all s: Segnalation | s in Ticket.segnalations iff s.takenCareOf=True
}
fact rightPersonBilled {
all t: Ticket| t.issuedTo=t.segnalations.vehicle.ownedby
}
fact noDoubleBilling {
all t1: Ticket| all s: Segnalation| s in t1.segnalations implies no t2:Ticket| t2 != t1 and s in t2.segnalations
}

ASSERTIONS
G1
assert eachSegnalationHasOneAndOnlyPhoto {
all disj s1, s2: Segnalation|
#s1.photo=1 and #s2.photo=1 and s1.photo != s2.photo and #Photo=#Segnalation
}
G2
assert dataMining {
```

```
all u: User| all ma: MunicipalAuthority|
#u.areaOfInterest>=0 and #ma.trackedAreas>=0 and
#ma.trackedUsers>=0 and #ma.trackedVehicles>=0
}
G3
assert everySegnalationHasOneAndOnlyPlate {
all disj s1, s2: Segnalation|
s1.writtenPlate=s2.writtenPlate iff s1.vehicle=s2.vehicle
}
G4
assert everySegnalationHasTimeAndPlace {
#s1.positionAndTime=1 and #s1.positionAndTime.coords=1 and #s1.positionAndTime.time=1 and
#s1.positionAndTime.coords.latitude=1 and #s1.positionAndTime.coords.longitude=1
}
GA1.2
assert interventionsSuggested {
all a: Area| #a.interventions>=0
}
GA2.4
assert ticketToVehicleOwner {
all v:Vehicle | all s: Segnalation | all t: Ticket|
(s in t.segnalations and v = s.vehicle) implies t.issuedTo=v.ownedby
}

WORLD
pred world1 {
#Vehicle=2
#Segnalation=3
#Ticket>0
#PositionAndTime>3
#Person>2
#Policeman>0
#Area>0
#MunicipalAuthority>0
#User>2
}

run world1 for 6
check eachSegnalationHasOneAndOnlyPhoto for 6
check dataMining for 6
check everySegnalationHasOneAndOnlyPlate for 6
check everySegnalationHasTimeAndPlace for 6
check intervetionsSuggested for 6
check ticketToVehicleOwner for 6
```

5 Effort Spent

Provide here information about how much effort each group member spent in working at this document. We would appreciate details here.

References

[1] S. Bernardi, J. Merseguer, and D. C. Petriu. A dependability profile within MARTE. *Software and Systems Modeling*, 10(3):313–336, 2011.