Politecnico di Milano
Dipartimento di Energia,
Informazione e Bioingegneria
Computer science and engineering

Software engineering 2

SafeStreets - DD

Professor:
Prof. Elisabetta Di Nitto

Candidates:
Nicolò Albergoni - 939589
Luca Loria - 944679

Academic Year 2019/2020
Milano - 09/12/2019

Contents

Introduction

1.1 Purpose

The purpose of this Design Document (DD) is to give a rather technical and implementation oriented perspective for the SafeStreets software that is going to be build. While the Requirement Analysis and Specification Document (RASD) gives a more conceptual description of the software and a view of the system that is not strictly related to the actual implementation, the DD provides a three hundred and sixty degree guide for the developers that will implement the software in the real world. In fact the DD document contains a practical and detailed description of the architecture that will have to be built; it covers all the aspects of the system that are relevant for the developers. For example the second section is entirely dedicated to the description of the system architecture in all the useful ways (i.e. component, deployment and runtime view), there is also a section for the design of user interface to be realized as well as a part related to the implementation and testing plan.

1.2 Scope

1.3 Definitions, acronyms and abbreviations

1.3.1 Definitions

- *Most Dangerous Streets*: Streets with the highest frequency of violation reports.

1.3.2 Acronyms

- EU: European Union;

- CET: Central European Timezone;
- API: Application Programming Interface;
- HTTP: Hyper Text Transfer Protocol;
- GPS: Global Positioning System;
- GDPR: General Data Protection Regulation.

1.3.3 Abbreviations

- [Gn]: n-th goal;
- [Dn]: n-th domain assumption;
- [Rn]: n-th functional requirements;
- MDS: Most Dangerous Street;
- LSA: Local System Administrator;
- PT: Police Technician.

1.4 Revision History

- Version 1.0:
  - First release.

1.5 Reference Documents

- SafeStreets assignment document;
- Previous years RASDs of the Software engineering 2 project;
- IEEE Std 830--1998 IEEE Recommended Practice for Software Requirements Specifications;
- Slides from the course "Software engineering 2".

1.6 Document Structure

1. In the first part a general introduction of the Design Document is given. The purpose part exposes the substantial differences with the RASD document;

2. The second section it's the core of the DD, it firstly provides a high level overview of the system followed by a description of various aspects of the architecture from different points of view such as: component, deployment and runtime view. It is also present a general explanation of the architectural patterns and styles adopted in the development process. Most of the parts of this section are enriched with UML diagrams to ensure a better understanding of the concepts;

3. This part specifies the user interface design of the mobile application and the web interface. Since the mockups of both applications were already inserted in the RASD document here are proposed some UX diagrams to better describe the navigation and functioning of the applications;

4. Part four exposes the requirements traceability matrix which maps the requirements stated in the RASD document with the corresponding design element;

5. Chapter five provides the proposals for the implementation, integration and testing plans. This plans are created by taking into account, for each functionality, the importance for the customer and the difficulty of implementation/testing;

6. The last part states the hours of work division and the tools used to create all the part of this DD document.

Architectural Design

2.1 Overview: High-level components and their interaction

2.2 Component view

2.3 Deployment view

The following image represents the deployment diagram of the SafeStreets architecture. It shows the logical division of the software architecture as well as the distribution of the software components to their target nodes, on which they will be deployed.

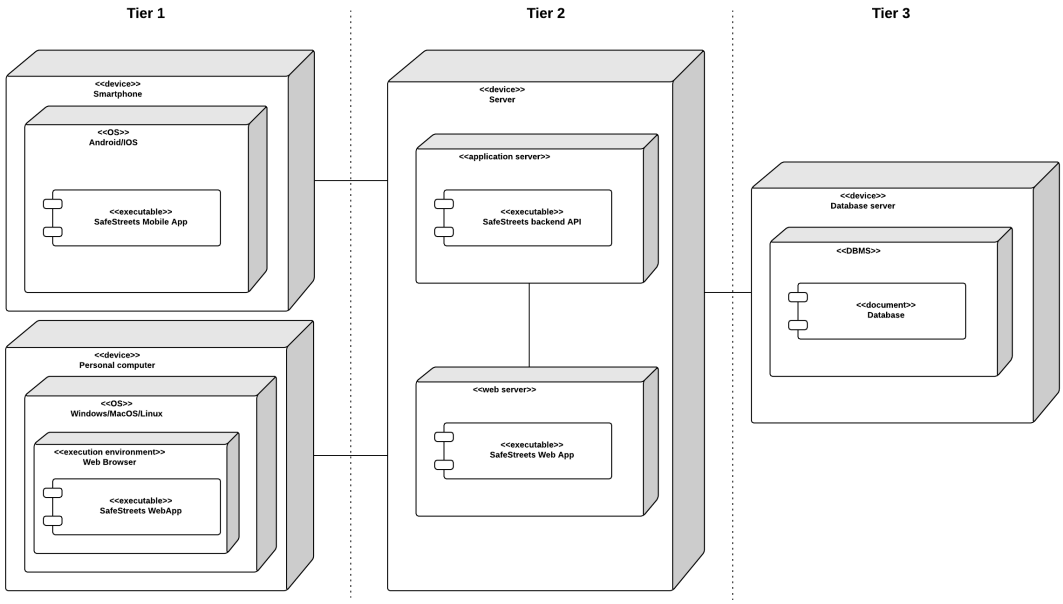

Figure 2.1: Deployment diagram

The diagram represent the architectural pattern chose for the deployment of the SafeStreets service. It's a three tier architecture that has been conceived in order to provide a fast and lightweight client for both the users and the authorities. It is also important to note that the previous diagram is focused only on those components that will have to be effectively developed,that's why, for example, the maps and the plate recognitions services are not displayed in the diagram, as they already exist. Here is a short explanation of what each tier does:

- **Tier 1**: The first tier is the presentation tier of the architecture. The main function of this layer is to host the the application of the users and the Web app of the authorities. It is worth noting that there's little or nothing business logic present in this layer apart from those that are strictly required for the correct functioning of the applications; the application and the Web App are only presentation client that will have to interact with the second tier in order to get/elaborate information. As for the deployment aspect: the mobile application must be developed in order to cover the most of the devices, so it must be runnable at least on Android and IOS (to reduce the development time one could also think to use a cross-platform development framework) while the Web app must be compatible with the most modern web browser such as: Edge, Chrome, Firefox and Safari. As stated before the applications in this tier are just thin clients so they need to interact with the application server in order to get information. The Mobile application asks the information directly to the application server via the backend API while the Web app communicate with the web server that will eventually ask to the application server for data. All the communications are performed with RESTful calls over a secure connection.

- **Tier 2**: The second tier of the architecture is called the logic layer, it is the place in which all the business logic is located; all the functionalities and the elaborations are computed here. The implementation is achieved via a server that runs two subservices separately: the Application server and the Web server. The Application server hosts the core of the system that is to say the SafeStreets Backend API, this is the piece of software that will handle all of the requests and the offered services. Instead the Web server hosts the content for the web app, in case that some page needs to be created with some data, the Web server can communicate with the

backend API in order to retrieve the information needed.

- **Tier 3**: The final layer is the so called Data tier, it handles the data access with the remote NoSQL database. The choice to move this into a separate tier is due to the fact that with this approach the data are independent from the business logic.

2.4 Runtime view

2.5 Component interfaces

2.6 Selected architectural styles and patterns

2.7 Other design decisions

User interface design

Requirements traceability

Implementation, integration and test plan

Effort Spent

6.1 Luca Loria

Day	Hours	Topic
20/10/2019	1.5	Text assumptions
22/10/2019	1	Domain assumptions
23/10/2019	2	Design constraints
24/10/2019	2	Revision ch 1 and 2
26/10/2019	3	Software system attributes
28/10/2019	0.5	Performance requirements
29/10/2019	2	External interface req
30/10/2019	5	Mockups creation
31/10/2019	1.5	Revision ch 3
01/11/2019	1	Alloy signatures
04/11/2019	1.5	Alloy facts part 1
06/11/2019	3	Alloy facts part 2 and world predicates
07/11/2019	4	Finish alloy, fix class diagram and start impaginating
08/11/2019	3.5	Reviewing requirements and sequence diagrams. Create references and Revision. Create alloy world section in RASD. Start impaginating correctly
09/11/2019	2	Shared phenomena matrix, frontpage and pagination fixes
10/11/2019	2	Final Revision

6.2 Nicolò Albergoni

Day	Hours	Topic
22/10/2019	2.5	Purpose
23/10/2019	3	Scope, Current system
24/10/2019	1.75	Goals, Overview
26/10/2019	3	Product perspective, Product function
27/10/2019	3	Product function, Revision ch 1 and 2
29/10/2019	2.5	Scenarios
30/10/2019	4	Use cases
31/10/2019	3	Use cases, Revision ch 3
01/11/2019	1	Finish use cases
02/11/2019	2.75	Requirements
05/11/2019	2.25	Finish requirements
06/11/2019	2.5	Sequence diagrams
07/11/2019	3.25	Sequence diagrams
08/11/2019	1.75	Finish and reviewing of sequence diagrams
09/11/2019	2	Use case diagrams, pagination fixes
10/11/2019	2	Traceability matrix, final revision

References

- LateX Workshop extension for Visual Studio Code:
  https://github.com/James-Yu/LaTeX-Workshop/
- LateX compiler:
  https://www.latex-project.org/
- StarUML for UML diagrams:
  http://staruml.io/
- Moqups for creating Mockups:
  https://app.moqups.com/
- Alloy extension for Visual Studio Code:
  https://github.com/s-arash/org.alloytools.alloy/tree/ls
- Alloy compiler and visualizer:
  http://alloy.lcs.mit.edu/alloy/