Politecnico di Milano
School of Industrial and Information Engineering
Master of Science in Computer Science and Engineering

Software Engineering 2 Mandatory Project

Safe Streets.

Design Document

Authors:
Rosetti Nicola
Sartoni Simone
Torri Vittorio

Academic Year 2019/2020

Milano, 09/12/2019
Version 1.0

Contents

List of Figures

List of Tables

1 Introduction

1.1 Purpose

1.2 Scope

1.3 Definitions,Acronyms,Abbreviations

1.4 Revision History

1.5 Reference Documents

1.6 Document Structure

2 Architectural Design

2.1 Overview: High-level components and their interaction

The application will be developed using the client-server paradigm on a three-tiered architecture. The three layers of the application (Presentation, Application and Data) are divided into clusters of machines (i.e. tiers) that actually cooperate to provide a specific functionality. In this case we have three tiers and each tier is responsible for one of the three layers. The client side is responsible (only) for the presentation layer; therefore, in this architecture, the thin-client has been adopted considering the fact that the required functionalities client-side are limited. The UIs provided are just meant to show results and to allow clients to choose what they want. In the App case, the client contains all the presentation layer while in the WebApp case the layer is splitted between the client and WebServer; the WebServer is responsible for contacting the application server and forward the client requests to it. The Application tier takes care of the application layer encapsulating all is needed concerning the application logic. It receives the requests from the clients and handles them. It's also responsible for sending asynchronous notifications to the presentation layer when certain conditions are met. Here we have multiple Servers cooperating together to improve performance, scalability, fault tolerance and availability. An elastic component (i.e. load balancer) is used to rule the accesses to different Application Servers, dinamically balancing the load among all the Servers. The Application tier communicates with the Data tier, responsible for the Data Access layer. This tier is composed by several DataServers: each one is associated with a single replica of the data and exploits the DBMS technology to access the DataBase. The Database is fully replicated in different nodes. Techniques and protocols are used to ensure consistency among replicas: they will be fullt explained in "Other design decisions" section. Again a load balancer is used to dinamically share the load among different machines. To ensure and improve security firewalls are installed before and after the application servers to filter accesses from external and unsafe networks. By the creation of a DMZ (demilitarized zone) external entities can only have access to the exposed services. Security is crucial because the application works mainly with sensible informations. To provide the required functionalities the system exploits datawarehousing. The datawarehouse is a component in the Data tier able to deal with historical data and aggregate data taken from the Databases exploiting data mining technologies to answer complex queries: used techniques are clustering, associative rules and classification. This component periodically queries one dataServer to retrieve new information and updates on the data since its last update.

2.2 Component view

2.3 Deployment view

2.4 Runtime view

2.5 Component interfaces

2.6 Selected architectural styles and patterns

2.7 Other design decisions

2.7.1 Consistency and update strategies among replicas

In this application, we need a client-centric consistency among replicas because end users (and therefore application servers that act as clients towards the DataServers) don't always connect to the same DataServer, due to the presence of the load balancer. Every DataServer can respond to a request to read or write data so we use an active replication protocol. In particular we exploit leaderless replication in which the decision on the value to read and the write to perform is decided by all the replicas or at least a quorum of them. The type of chosen consistency model is the "read your writes": the effect of a write operation by a process on a data item x will always be seen by a successive read operation on x by the same process. Concerning the update propagation we opted for propagating a notification of the operation, assuming that there will be more writes than reads. The propagation strategy chosen is the Gossiping strategy: when a replica is updated then it just propagates that update to all the nodes that it knows; if a replica receives an update that it has already received then the probability of propagating that information is decreased on that replica.

3 User interface design

4 Requirements Traceability

- [R1] The system must allow people to register to it providing personal data (name, surname, birthdate, identity card number, fiscal code) and selecting a username and a password.
- [R2] The system must verify the correctness of the provided personal data of a registered user checking them from the identity card number, blocking the registration if they are not correct.
- [R3] The system must allow registered users to login through their username and password.
- [R4] The system must allow logged user to fill a report violation form.
- [R5] The system must let the user select the type of violation detected.
- [R6] The system must allow the user to insert the license plate in a violation report.
- [R7] While reporting the violation, the system must allow users to take one or more pictures of the potential violation.
- [R8] The system must not allow users to choose pictures not taken in the moment of the report.
- [R9] The system must collect the current position of the user, using GPS.
- [R10] The system must allow user to confirm or delete the current report.
- [R11] After confirmation, the system must add the current date and time to the report.
- [R12] The system must store confirmed report.
- [R13] The system must check reports to try to find if the pictures of the violations have been modified.
- [R14] The system must try to find, according to the GPS position of the user and the pictures sent, if the position is fake or not.
- [R15] The system must discard the report if it has been recognized as fake according to the previous requirements (R13-R14).
- [R16] The system must try to automatically recognize the license plate in the photo, possibly with the help of the value inserted by the user.
- [R17] The system must store into stable memory the reported violation if correct (i.e. not recognized as fake).

- [R18] The involved municipality must be calculated considering in which city the reported violation has been found, based on the GPS position of the user that has sent the report.
- [R19] The system must send the reported violations to the involved municipality.
- [R20] The system must allow an agent to see the reports for its municipality, checking them in order of arrival
- [R21] The system must allow an agent to issue a traffic ticket to a certain person (i.e. license plate) through the correspondent municipality service.
- [R22] The system must allow an agent to put on hold a violation report if it needs to be checked in person.
- [R23] The system must allow an agent to discard a violation if it has been verified as fake or it cannot be verified (the vehicle is not there anymore) or it is a duplicated report.
- [R24] The system must allow an agent to retrieve the data of the author of a violation report.
- [R25] The system must allow an agent to create an account, asking the municipality services to verify its identity.
- [R26] The system must allow an agent to login, inserting its username and password.
- [R27] The system must mine this information from the reported violations.
- [R28] The system must allow the users (even if not authenticated) to select the see information for a city. The user can choose either the city where he is, using the GPS position, or an arbitrary selected location.
- [R29] The system must allow the user to select information about streets or areas in the city selected and to specify if he wants information for a specific street or area or a classification of streets or areas.
- [R30] The system must show the the data corresponding to the selection of [R29]
- [R31] The system must take information about accidents and tickets from the municipality.
- [R32] The system must use this information to build statistics, crossing them with reported violations.
- [R33] The system must not allow common users to see confidential data about other people.
- [R34] The system must allow the user to choose a topic: areas or streets with most accidents, areas or streets with the highest number of traffic tickets issued, areas or streets where there have been the best improvements, information for a specific area or street.
- [R35] The system must show to the user the information about the topic selected according to [R34].
- [R36] The system must allow authenticated supervisors to retrieve information about the vehicles with the highest number of violations in a selected area or street.
- [R37] The system must allow supervisors to access only information about their own municipality.

- [R38] The system must allow a supervisor to create an account, asking the municipality services to verify its identity.
- [R39] The system must allow a supervisor to login, inserting its username and password.
- [R40] The system must take information about accidents, tickets and street networks (bike lanes, sidewalks, parking areas,...) from the municipality, exploiting the municipality services.
- [R41] The system must elaborate this information, combined with reports information, and try to find possible solutions for problems.
- [R42] The system must notify the municipality

5 Implementation, integration and test plan

6 Effort spent

Nicola Rosetti

Date	*Hour*	*Section*
17-10-2019	1.5 h*	Goals

Table 6.1

Simone Sartoni

Date	*Hour*	*Section*
17-10-2019	1.5 h*	Goals

Table 6.2

Vittorio Torri

Date	*Hour*	*Section*
18-11-2019	1 h*	Goals

Table 6.3

** Group work*

7 References