POLITECNICO
MILANO 1863

Politecnico di Milano
A.A. 2016–2017
Software Engineering 2: "PowerEnJoy"
Design Document

Pietro Ferretti, Nicole Gervasoni, Danilo Labanca

December 1, 2016

Contents

1 Introduction

1.1 Purpose

"The purpose of this document is to provide a complete and detailed description and specification of a digital management system for PowerEnJoy, an electric car sharing service. This document will illustrate functional and non-functional requirements of the software to-be, outlining costraints, showing potential user interfaces for the software, and explaining the domain assumptions made. Additionally, at the end of the document we will present an Alloy model to further specificate the world and environment our system will have to manage."

1.2 Scope

text here

more text here

....

1.3 Definitions, Acronyms, Abbreviations

1.3.1 Definitions

- *Guest*: a person that is not registered to the system.
- *User*: a person that is registered to the system. Users can log in to the system with their email or username and their password. Their first name, last name, date of birth, driving license ID are stored in the database.
- *Safe area*: a location where the user can park and leave the car. Users can end their ride and park temporarily only in these locations. The set of safe areas is predefined by the system.
- *Power grid station*: a place where cars can be parked and plugged in. While a car is plugged in a power grid station its battery will be recharged. Power grid stations are by definition safe areas.
- *Available car*: a car that is currently not being used by any user, and has not been reserved either. Available cars are in good conditions (not dirty nor damaged) and don't have dead batteries.
- *Reservation*:
  - the operation of making a car reserved for a user, i.e. giving permission to unlock and use the car only for that user, forbidding reservations by other users.
  - the time period between the moment a reservation is requested and the moment the user unlocks the car, or the reservation is canceled.
- *Ride*: the time period from the moment a reserved car is unlocked to the moment the user notifies that he wants to stop using the car and closes all the doors. A ride doesn't stop when a car is temporarily parked, but continues until the user chooses to leave the car definitely.
- *Possession*: users that have reserved and unlocked a car are said to have possession of the car. While a user has possession of a car they are the only person that can drive it, lock or unlock it, and no other person can take possession of it until the user frees it. Users lose possession of a car when their ride ends.
- *Temporary parking*: the act of parking a car in a safe area and, after notifying the system, locking it and leaving it for a finite amount of time. The user that does this retains the right to use the car and can unlock it later to use it again.
- *Bill*: a record of the money owed by the user at the end of a ride.
- *Outstanding bill*: a bill that hasn't been paid yet.

- *Suspended user*: a user that cannot reserve or use cars. Usually users are suspended because they have outstanding bills.
- *Payment method*: a way to transfer money from the user to the system. Our system will only accept credit cards and online accounts like Paypal.
- *Payment API*: an interface to carry out money transactions, offered by the external provider associated to the payment method used (e.g. a bank).
- *CAN bus*: a vehicle bus standard designed to allow micro controllers and devices to communicate with each other.

1.3.2 Acronyms

- **DD**: Design Document
- **RASD**: Requirements Analysis and Specification Document
- **DB**: Database
- **CVV**: Card Verification Value
- **DOB**: Date of birth
- **PGS**: Power Grid Station
- **GPS**: Global Positioning System
- **CAN bus**: Controller Area Network bus

1.3.3 Abbreviations

- **[Gx]**: Goal
- **[RE.x]**: Functional Requirement
- **[UC.x]**: Use Case

1.4 Reference Documents

- IEEE Std. 1016-2009, "IEEE Standard for Information Technology – Systems Design – Software Design Descriptions"
- ISO/IEC/IEEE Std. 42010:2011, "Systems and software engineering – Architecture Description"
- Specification document: "Assignments AA 2016-2017.pdf"

1.5 Document Structure

This document is structured as follows:

- **Section 1 – Introduction:** this section introduces the design document. It contains a justification of his utility and indications on which parts are covered in this document that are not covered by the RASD.
- **Section 2 – Architectural Design:** this section is divided into more parts:
  - **Overview:** this section shows a high-level view of our system's architecture.
  - **Component View:** this section gives a more detailed (in-depth) view of the components of the application and how they communicate.
  - **Deployment View:** this section shows the way the components will be deployed to run our application.
  - **Runtime View:** in this section we use sequence diagram to show the flow of the most important functionalities of the system.
  - **Component Interfaces:** the interfaces between the components are presented in this section.
  - **Selected Architectural Styles and Patterns:** this section explains the architectural choices taken during the creation of the application.
  - **Other Design Decisions**
- **Section 3 – Algorithm Design:** this section describes the most critical parts via some algorithms. Pseudo code is used in order to hide unnecessary implementation details in order to focus on the most important parts.
- **Section 4 – User Interface Design:** this section presents mockups and user experience explained via UX and BCE diagrams.
- **Section 5 – Requirements Traceability:** this section aims to explain how the requirements identified in the RASD are linked to design elements.
- **Effort Spent**
- **References**
- **Revisions**

2 Architectural Design

2.1 Overview

2.2 Component View

2.3 Deployment View

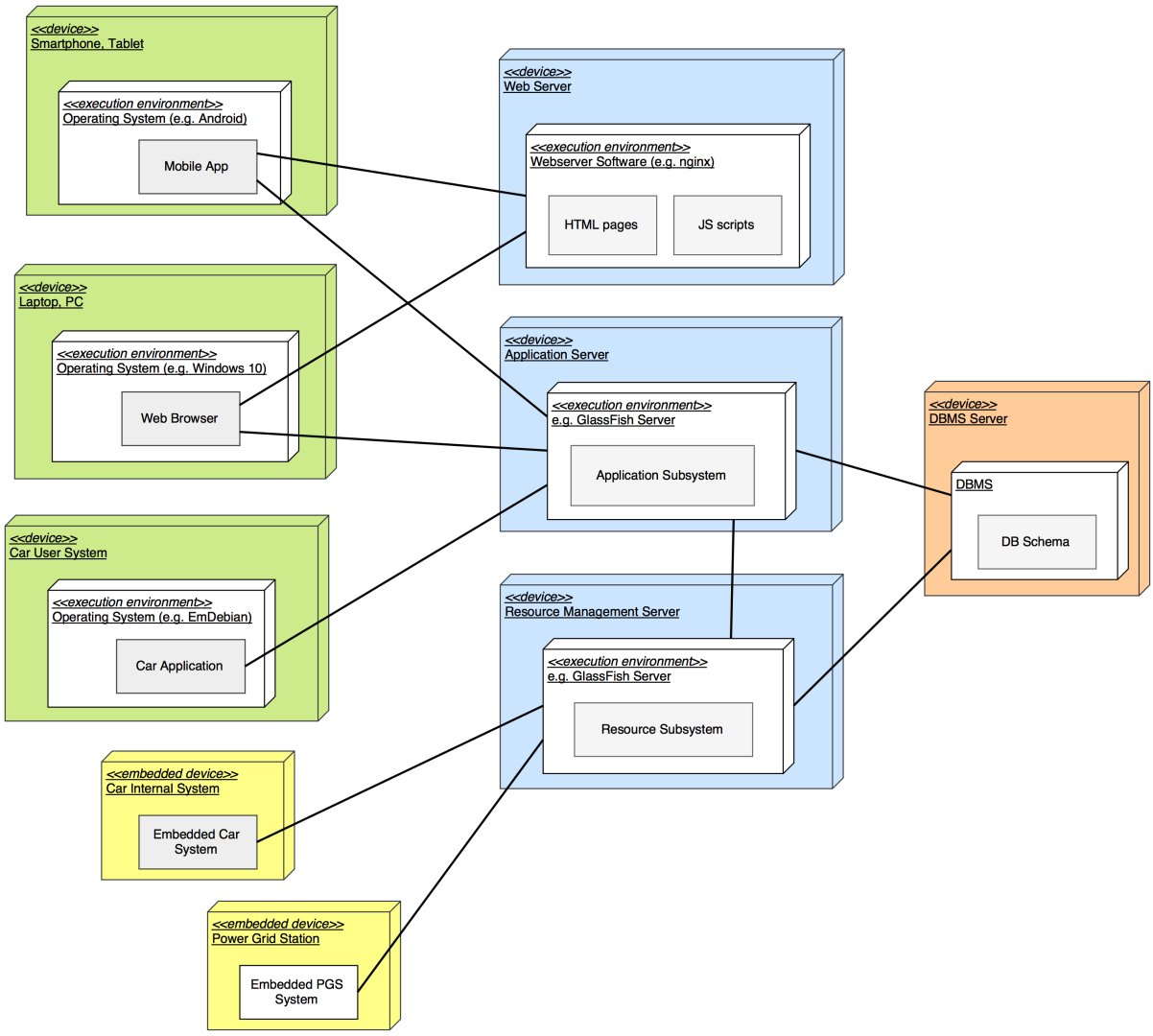

Figure 1: Deployment Diagram

tre tier: client, business logic, database

client: - app e browser si collegano al web server per avere le pagine web, poi si collegano all'application server per raccogliere i dati - l'applicativo sulla macchina non è basato sul web e si collega direttamente all'application server per avere tutto - esiste un "resource management server" che raccoglie informazioni (gps, ecc.) dalle macchine e disponibilità posti dalle PGS - l'application server inoltre può inviare comandi alle macchine tramite il resource server - application

server e resource server sono collegati al DBMS che contiene tutto in tempo reale

2.4 Runtime View

2.5 Component Interfaces

2.6 Selected Architectural Styles and Patterns

2.6.1 Protocols

2.6.2 Patterns

2.7 Other Design Decisions

3 Algorithm Design

4 User Interface Design

4.1 Mockups

4.2 UX Diagrams

4.3 BCE Diagrams

5 Requirements Traceability

6 Effort Spent

7 Revisions

7.1 Changelog

-