Politecnico di Milano

Software engineering 2

PowerEnjoy
ITPD v1.0

Alessandro Caprarelli	874206
Roberta Iero	873513
Giorgio De Luca	875598

January 12, 2017

Contents

Chapter 1

Introduction

1.1 Revision history

Table 1.1: Revision history

Version	Date
v1.0	15/01/2017

1.2 Purpose

This document represents the Integration Test Plan (ITP) and its main purpose is to describe the tests that we need to perform in order to verify that all the components of the software, previously identified in the Design Document, are correctly linked and work together. This document not only introduces the tests to be performed, but it also reasons the decision to take this set of tests, the order in which they will be executed and the expected result of each of them. In this step of the software design we consider components as black-boxes and the goal is to achieve a correct interoperability of them.

1.3 Scope

The aim of the project PowerEnjoy is to provide an automated service of car sharing. After a registration, a client can hire a car near him/her through the web or mobile

application and he/she can enjoy of all the extra services offered. The exact position of the client is determined by the GPS signal of the client's device or it's allowed to manually insert a specific address. So the system displays all the available cars in the client's close area. Then the client could make a reservation of a car, after which the system notifies the client with a message of confirmation with the car identifier. If the reservation procedure successfully ends the chosen car won't be available anymore for other clients. Moreover a client cannot hire more than one car at the same time. After the reservation the client has at most one hour to reach the car, when this time expires the system gives a penalty to the client and the car, previously hired, is available again for other clients. The system allows the client to cancel his/her reservation. When the client reaches the car, he/she can tell the system that is nearby through a specific button in the application and he/she starts to pay as soon as the engine ignites. During the travel the system supervises the current charge of the car and notifies it to the client through a screen located in the car. The system stops charging the amount of money that the client has to pay when he/she communicates through the application his/her decision to stop the rent. When the car is parked in a safe area and the client exits, the system locks the car automatically and starts the procedure of payment. The client is notified with the result of this procedure through an SMS, including the final fare.

1.4 Definition, acronyms, abbrevations

1.4.1 Definition

- **Guest client:** a person that is not already registered in the system or that has to log in.
- **Registered client:** a person who has valid access credentials to log in the system.
- **System administrator:** privileged user, in charge of managing administration processes and of updating business logic.
- **Reservation:** it is the action performed by a registered client that allow him/her to reserve an available car for maximum one hour.
- **Journey time = travel time:** time elapsed since the user starts the engine to the user parks the car and terminates the journey.
- **Available car:** a car that is not reserved by any user and has enough charge to be rented.

- **Unavailable car:** a car that is already reserved or damaged, so impossible to reserve.
- **Gps navigation:** it is the navigation system that is included in the car on board system. It could be used by the user to find direction to the final destination.
- **Final destination:** address where the user wants to go.
- **Safe area:** the region where is permitted to park and leave a car once the rent is terminated.
- **Power grid station:** the area where it's allowed users to park the cars, leaving them attached to the power grid.

1.4.2 Acronyms

- **ITP:** Integration Test Plan
- **DD:** Design Document
- **RASD:** Requirements Analysis and Specification Document
- **API:** Application Programming Interface
- **UI:** User Interface
- **DBMS:** Data Base management system

1.4.3 Abbrevetations

- **[In.m]:** //TODO

1.5 Reference documents

- RASD v1.1
- DD v1.0
- PowerEnjoy specification document (assignment).
- IEEE Std 1016tm-2009 Standard for Information Technology - System Design - Software Design Descriptions.

1.6 Document overview

The ITP is composed of five sections:

- Introduction: it provides a general description of the content of the Integration Plan Document and some additional information in order to be coherent with the RASD document and the Design Document, previously produced.
- Integration Strategy: this section introduces the strategy used for the integration, motivates the choices made and explains also the approach used to integrate. In fact the sequence to follow to perform the integration tests is specified.
- Individual Steps and Test Description: in this part all the test sets are defined and described according to the choices made in the previous section. In particular the expected result of each test case is specified.
- Tools and Test Equipment Required: this section contains a list of all the software tools and test equipment needed to perform the integration tests.
- Program Stubs and Test Data Required: in the last part of the document all the program stubs and all the special test data required for the integration test phase are identified and characterized.

Chapter 2

Integration strategy

This section describes decisions and criteria chosen for the ITP of the system of "PowerEnJoy".

2.1 Entry criteria

Before starting the Integration Testing phase, it is necessary to have successfully completed the Unit Testing of all the modules of the software. So from now on we assume that all the single modules of the system of PowerEnJoy work correctly. In addition to that we need some prerequisites to be satisfied:

- The software of PowerEnJoy must be code-completed.
- The system must satisfy all the requirements of the RASD and all the decisions taken in the DD.
- The Database should be ready and its tables are populated with initial data.

2.2 Elements to be integrated

The following image represents the Component View of the system of PowerEnJoy, as it has been designed in the DD. We bring it here again because it clearly shows all the components of Power EnJoy that are going to be tested for integration and all the dependences existing between them. These dependencies are particularly important because they give us a guideline to follow identifying the Integration Testing Strategy.

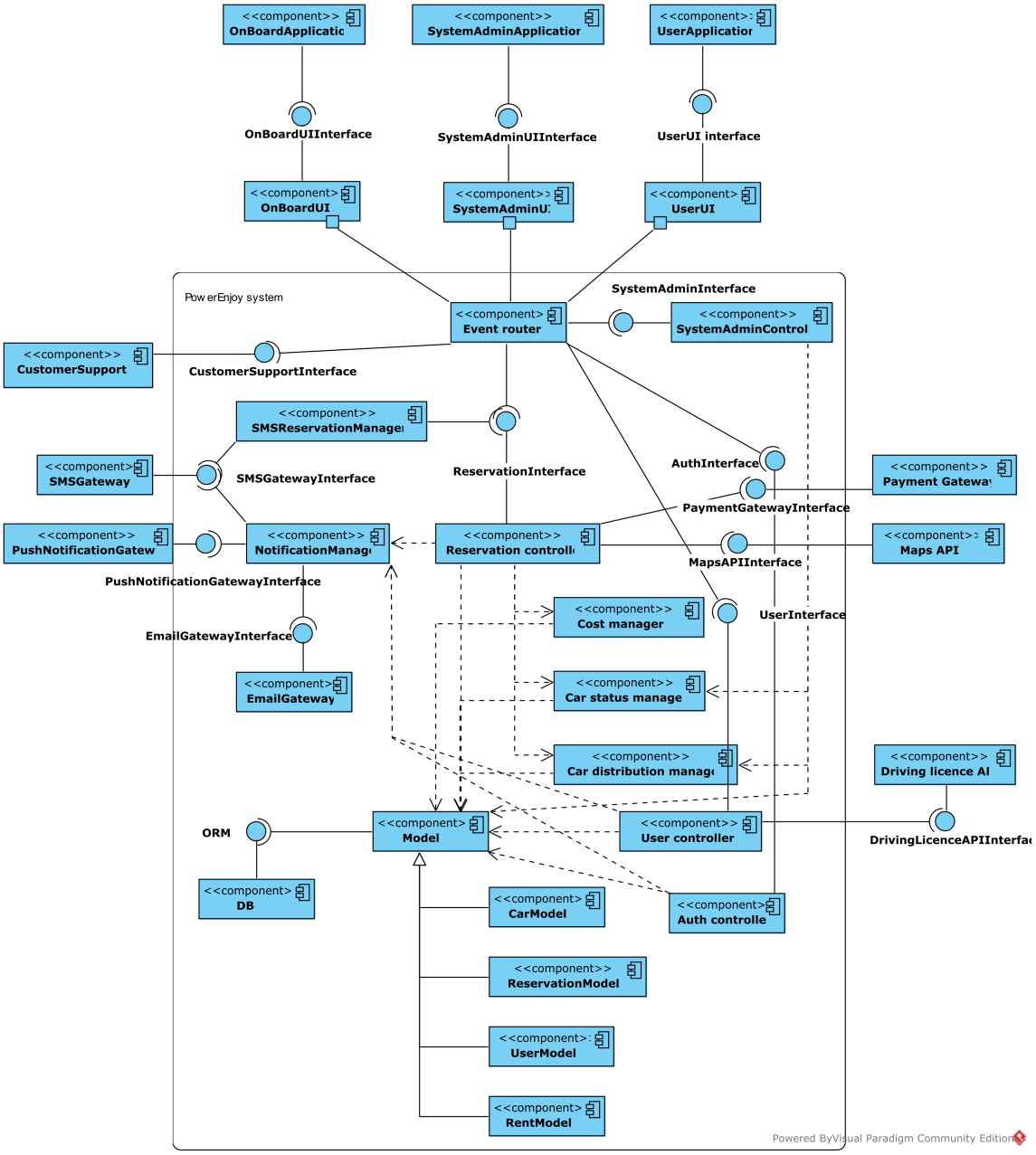

Figure 2.1: Component diagram

2.3 Integration Testing Strategy

We decide to adopt a Bottom Up approach in order to test the integration of modules of the system of PowerEnJoy. Bottom-up testing is a type of integration testing that integrates code units and tests them together, before testing the entire system. In this approach testing is conducted from sub module to main module, if the main module is not developed a temporary program called 'driver' is used to simulate the main module. For example if we look at the following figure, first of all we test module 4, 5, 6 and 7 individually using drivers. Then we test module 2 such that it calls 4 and 5 separately. If an error occurs we know that there is a problem in one of the modules, or in the interface between them. Then we test module 1 such that it calls module 3 and again if an error occurs we know that the problem is in module 3 or in the interface between module 1 and module 3.

//IMMAGINE BOTTOM-UP

2.4 Sequence of Component/Function Integration

The bottom up strategy provides a simple workflow to test modules and makes easier the formulation and the observation of tests. Since the application has been developed, it is possible to run tests returning on the tree of modules, without creating fake contexts of execution to support the tests.

This document wants to offer a testing plan based on a particular sequence to follow, explained in detail in the next paragraph.

2.4.1 Software Integration Sequence

The sequence of integration tests depends on the strategy used to verify the correctness of tests. In this case, the bottom up strategy has the advantage of providing simple creation of conditions of the test.

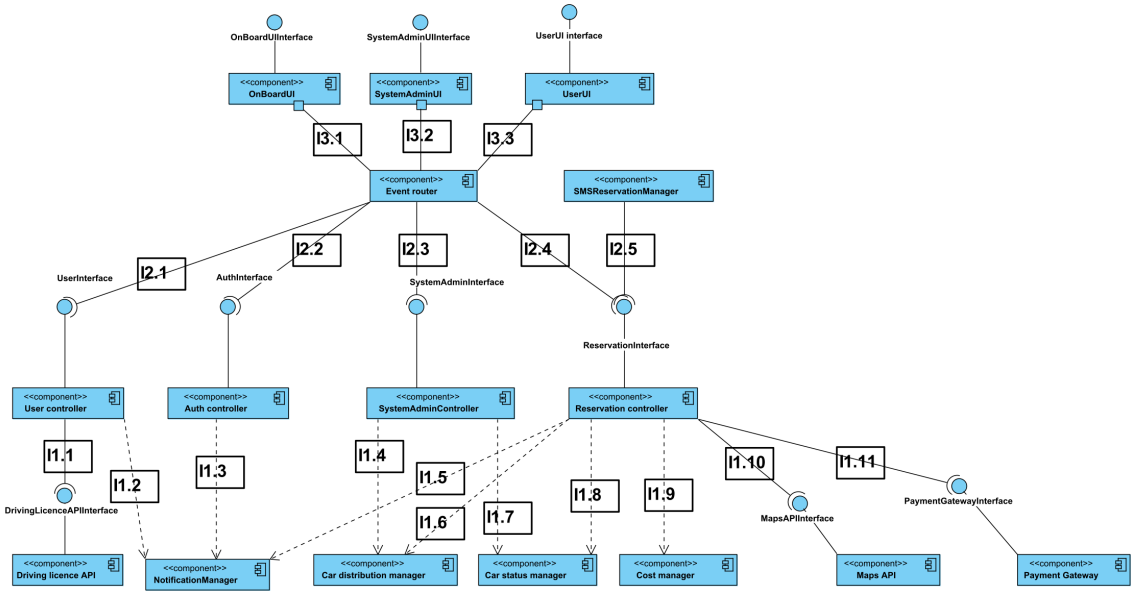

Figure 2.2: Integraton tests sequence

The order to follow is based on the type of component we are going to test:

- **Manager components** are the most important to be tested, because they represent the logic of the application; managers must be tested with specific inputs and various test environments to guarantee as much as possible the coverage of all the cases of the domain. It's suggested to start tests from easier functions/processes to more complex ones: in this way the next-level test can be run with no prerequisites on the atomic actions required.
- **Gateway components** are tested when the model and the controllers work as expected. Calls to external services could require an answer to the previous request: the services we call have a particular service for tests and return us always a positive answer. Processes at this point involve an important number of components: as before, start from easier to more complex ones.
- **UI components** are the last type of component to be tested because if each test has been positively passed before, we expect to have no surprise in the presentation of processed data. The order to follow is the same of the workflow present in the BCE.

Chapter 3

Used tools and working hours

3.1 Used tools

The tools used to create this document are the following:

- MikTex 2.9: to format the document using LaTex.
- Pencilç to create mookups.
- Visual paradigm: to create sequence diagrams.
- LucidChart: to create use case, state chart and class diagrams.
- Alloy Analizer 4.2: to realize the model and to check its consistency

3.2 Working hours

	Alessandro	Roberta	Giorgio
28/10	1.5	1.5	1.5
29/10		0.5	
30/10	2		1
31/10	1.5	3	2.5
1/11	0.5	0.5	4
2/11	0.5	1	2.5
3/11	3	4	2
4/11	3	2	
5/11		3	3
6/11	1.5	3	
7/11	3.5	1.5	1.5
8/11	2	3	2
9/11	1.5		2
10/11	0.5		
11/11	3	3	3
12/11	3	3	3
13/11	1	1	1
Total	28	30	29