Power EnJoy *Integration Test Plan Document*

Niccolo' Raspa, Matteo Marinelli

December 29, 2016

Software Engineering 2 Course Project

ASSIGNMENT INFO

The Integration Test Plan Document (ITPD) aims at describing how you plan to accomplish the integration test. This document is supposed to be written before the integration test really happens. Often it is written in parallel to the Design Document and takes the architectural description of the software system as a starting point. This document needs to explain to the development team what to test, in which sequence, which tools are needed for testing (if any), and which stubs/drivers/oracles need to be developed. The structure we suggest for this document is the following (if you introduce changes to this structure, please provide a justification for this):

Contents

1 Introduction

1.1 Revision History

Version	Date	Author(s)	Summary
1.0	06/01/2016	Niccolo' Raspa, Matteo Marinelli	Initial Release

1.2 Purpose and Scope

This document represents the Integration Testing Plan Document for PowerEnJoy. The main purpose of this document is to outline, in a clear and comprehensive way, the main aspects concerning the organization of the integration testing activity for all the components that make up the system.

This process is essential because it not only guarantees that every component behaves as expected but also that all the components interoperate correctly together to fullfil all the functionalities expected from the system.

We will focus deeply on the Application Layer since it implements all the core of our business. This component is divided in different services which provides great benefits but the implicit granularity of a service oriented approach must be fully tested to guarantee that all the subcomponents behave as one cohesive layer.

This document is structured as follows:

Chapter 1 Provides general information about the ITPD document.

Chapter 2 Explains in details the chosen integration strategy. In more details:

- Lists of the subsystems and their subcomponents involved in this process
- Specifies the criteria that must be met before integration testing begins
- Describes the integration testing approach and the rationale behind it
- Outlines the order in which components and subcomponents will be integrated

Chapter 3 Describes the type of tests that will be used to verify that every step of the integration process above perform as expected.

Chapter 4 Identifies all tools and test equipment needed to accomplish the integration.

Chapter 5 Identifies any program stubs or special test data required for each integration step

1.3 List of Definitions and Abbreviations

DD: Design Document

RASD: Requirement Analysis and Specification Document

ITPD: Integration Test Plan Document

EJB: Enterprise JavaBeans

SOA: Service Oriented Architecture

Component: One of the four tier of the system (Client, Web, Application, Database)

Subcomponent: Usually refers to the Application Layer, and refers to a EJB that encapsulates a specific part of the business logic of the module

Layer: synonim of *Component*

Service: synonim of *Subcomponent*

1.4 List of Reference Documents

Please refer to the following documents, for additional informations on the Power Enjoy System:

- Project rules of the Software Engineering 2 project
- Power Enjoy - Requirement Analysis and Specification Document
- Power Enjoy - Design Document
- **Documentation of any tool you plan to use for testing**

2 Integration Strategy

2.1 Entry Criteria

This section describes the prerequisites that need to be met before integration testing can be started.

Stakeholder Approval First of all, the Requirements Analysis and Specification Document and the Design Document must have been presented to the stakeholders for approval even before the coding phase can begin, this will ensure that they're satisfied with the development.

Website and Mobile App The presentation layer to the user might not be completed but communication between the Application Server and Clients, both via the Mobile App and via the Web Server, must have clearly structured and coded via RESTful APIs using JAX-RS.

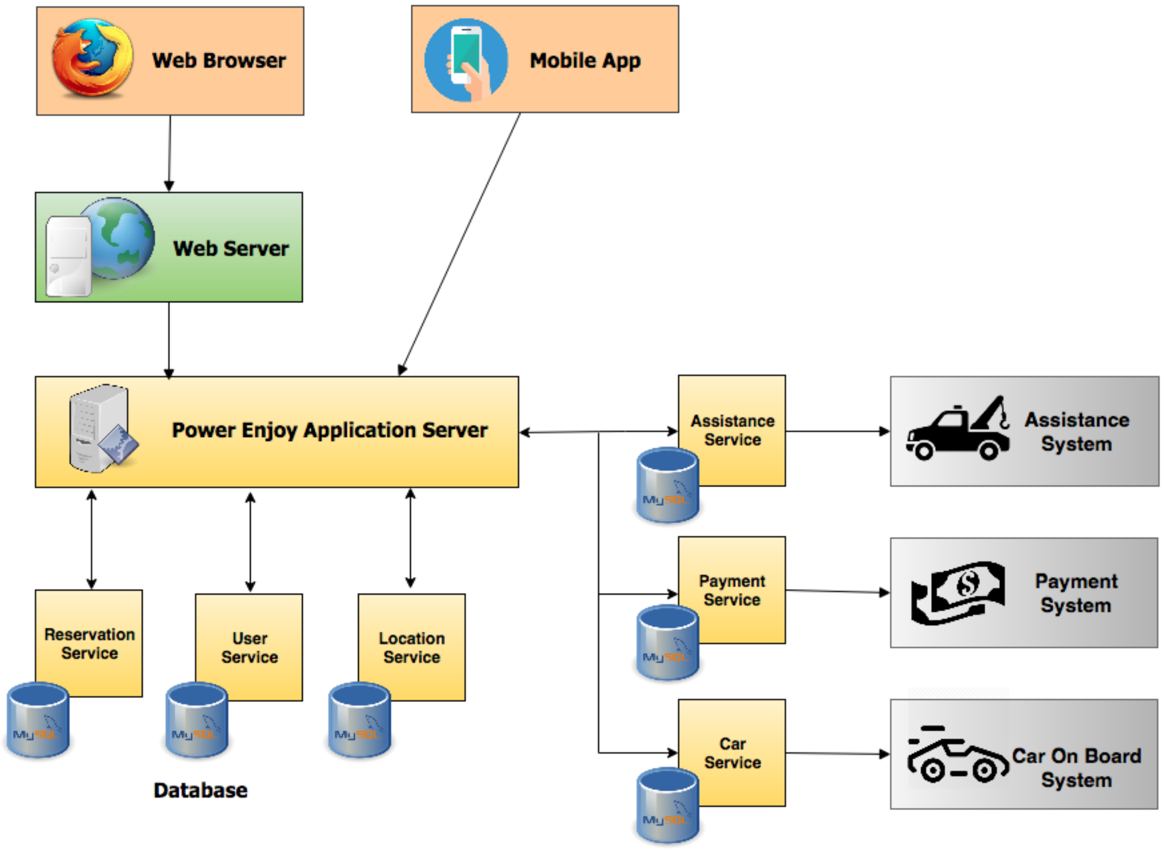
Coding and Testing Application Layer

- All the classes and methods of the Entity Beans must be coded and must pass thorough **Unit tests**. Unit tests should have a minimum coverage of 90% of the lines of code and should be run automatically at each build using JUnit.
- **Code inspection** has to be performed on all the code in order to ensure maintainability, respect of conventions and find possible issues which could increase the testers' effort in next testing phases. Code inspection should be performed using automated tools when possible.
- **Documentation** of all classes and functions should be well written using JavaDoc. The public interfaces of each Entity Bean shoud be clearly stated.
- **Object relational mapping** of each service to its corresponding database should be implemented with automated tools to avoid errors (such as Hibernate) and fully tested.

2.2 Elements to be Integrated

In this section we're going to provide a list of all the components that need to be integrated together. The figure below corrispond to the system architecture already discussed in the Design Document.

The system is built upon the interactions of many tiers, each one implementing a specific set of functionalities. Every tier is also obtained by the combination of several lower-level components. This modularity causes that the integration phase will involve the integration of components at different levels of abstraction. In addition our system is in relation with External Services, which are crucial for our application to work. It's important that they're correctly integrated to the system in a way in which everything is trasparent to the user.



In summary, the elements that needs to be integrated are:

1. Integration of the different subcomponents (classes, Java Beans) inside the same tier.
2. Integration of different tiers (Client - Web - Application)
3. Integration and configuration with third party services (Payment Service, Assistance Service)

2.3 Integration Testing Strategy

Describe the integration testing approach (top-down, bottom-up, functional groupings, etc.) and the rationale for the choosing that approach.

2.4 Sequence of Component/Function Integration

NOTE: The structure of this section may vary depending on the integration strategy you select in Section 2.3; use the structure proposed below as a non mandatory guide

2.4.1 Software Integration Sequence

For each subsystem, identify the sequence in which the software components will be integrated within the subsystem; relate this sequence to any product features that are being built up.

2.4.2 Subsystem Integration Sequence

Identify the order in which subsystems will be integrated; if you have a single subsystem, 2.4.1 and 2.4.2 are to be merged in a single section. You can refer to Section 2.2 of the test plan example [1] as an example

3 Individual Steps and Test Description

For each step of the integration process above, describe the type of tests that will be used to verify that the elements integrated in this step perform as expected. Describe in general the expected results of the test set. You may refer to Chapter 3 and Chapter 4 of the test plan example [1] as an example of what we expect. (NOTE: This is not a detailed description of test protocols. Think of this as the test design phase. Specific protocols will be written to fulfill the goals of the tests in this section.

4 Tools and Test Equipment Required

Identify all tools and test equipment needed to accomplish the integration. Refer to the tools presented during the lectures. Explain why and how you are going to use them. Note that you may also use manual testing for some part. Consider manual testing as one of the possible tools you have available.

5 Program Stubs and Test Data Required

Based on the testing strategy and test design, identify any program stubs or special test data required for each integration step.

6 Effort Spent

The approximate number of hours of work for each member of the group is the following:

Niccolo' Raspa	x Hours
Matteo Marinelli	y Hours