

Luca Scannapieco - 877145

Andrea Pasquali - 808733

Emanuele Torelli - 876210

15/01/2017

PowerEnJoy

Integration Test Plan Document

Table of contents

[1. Introduction 3](#_Toc471074781)

[1.1. Purpose and scope 3](#_Toc471074782)

[1.2. List of definitions and abbreviations 3](#_Toc471074783)

[2. Integration strategies 3](#_Toc471074784)

[2.1. Entry criteria 3](#_Toc471074785)

[2.2. Elements to be integreated 3](#_Toc471074786)

[2.3. Integration test strategies 3](#_Toc471074787)

[2.4. Sequence of component/function integration 3](#_Toc471074788)

[2.4.1. Software integration sequence 3](#_Toc471074789)

[2.4.2. Subsystem integration sequence 3](#_Toc471074790)

[3. Individual steps and test description 3](#_Toc471074791)

[4. Tools and test equipment required 3](#_Toc471074792)

[5. Program stubs and test data required 3](#_Toc471074793)

[6. Other info 3](#_Toc471074794)

[6.1. Reference documents 3](#_Toc471074795)

[6.2. Used tools 3](#_Toc471074796)

[6.3. Hours of work 3](#_Toc471074797)

[6.4. Changelog 3](#_Toc471074798)

# 1. Introduction

## 1.1. Purpose and scope

## 1.2. List of definitions and abbreviations

# 2. Integration strategies

## 2.1. Entry criteria

## 2.2. Elements to be integrated

## 2.3. Integration test strategies

We are going to use an incremental approach for integration testing. In particular, we will adopt essentially a bottom-up strategy with few slight modifications.

We will use the a purely bottom-up approach in order to build the component called “server” in the high level component diagram (DD chapter 2.2), that in essence represent the business layer of our application. Therefore, we will start integrating together the atomic subsystems of the server, i.e. the lower level components that do not depend on other components; then we will incrementally integrate the other subsystems that only depends on already integrated and tested components. This strategy, based on the hierarchical structure of the system, allows us to perform the integration test following the development process: as soon as components are released, we integrate them and test the integration. Furthermore, using bottom-up strategy for the server we reduce the overhead time needed to build stubs.  
In order to choose what to integrate among the atomic components we will follow the critical-module-first policy. In our case the most critical modules are the most used-one, such as the model that is the core of our MVC in the server side and therefore also the first component to be developed.   
For what concern the client side, we can say that we violate a bit the bottom-up strategy rules. In fact we are going to test the client side components such as Car App, Client App and Assistance Coordinator Program together with the server components even if the client side components use those of the server. This little modification of the strategy has the purpose of increase the parallelism of the work and consequently even the efficiency.

## 2.4. Sequence of component/function integration

### 2.4.1. Software integration sequence

### 2.4.2. Subsystem integration sequence

# 3. Individual steps and test description

# 4. Tools and test equipment required

# 5. Program stubs and test data required

# 6. Other info

## 6.1. Reference documents

## 6.2. Used tools

## 6.3. Hours of work

## 6.4. Changelog