

Version: 1.0.0

Release date: 16-12-2016

Riccardo Cattaneo 873647

Fabio Chiusano 874294

PowerEnJoy, ITPD

Table of Contents

[1 Introduction 3](#_Toc469679469)

[1.1 Revision History 3](#_Toc469679470)

[1.2 Purose and Scope 3](#_Toc469679471)

[1.3 List of Definitions and Abbreviations 3](#_Toc469679472)

[1.4 List of Reference Documents 3](#_Toc469679473)

[2 Integration Strategy 3](#_Toc469679474)

[2.1 Entry Criteria 3](#_Toc469679475)

[2.2 Elements to be integrated 3](#_Toc469679476)

[2.3 Integration Testing Strategy 3](#_Toc469679477)

[2.4 Sequence of Component – Function Integration 3](#_Toc469679478)

[2.4.1 Software Integration Sequence 3](#_Toc469679479)

[2.4.2 Subsystem Integration Sequence 3](#_Toc469679480)

[3 Individual Steps and Test Description 3](#_Toc469679481)

[4 Tools and Test Equipment Required 3](#_Toc469679482)

[5 Program Stubs and Test Data Required 3](#_Toc469679483)

[6 Effort Spent 3](#_Toc469679484)

# Introduction

## Revision History

### Version 1.0

The first release, it is published online before the deadline.

## Purpose and Scope

This Integration Test Plan Document aims at pointing out how to accomplish integration tests. Developers, testers and, in general, all the people involved in the development of the PowerEnjoy System should read this document before starting testing of the integration of components.

This document needs to explain to the development team what to test, in which sequence, which tools are needed for testing, and which stubs/drivers/oracles need to be developed.

## List of Definitions and Abbreviations

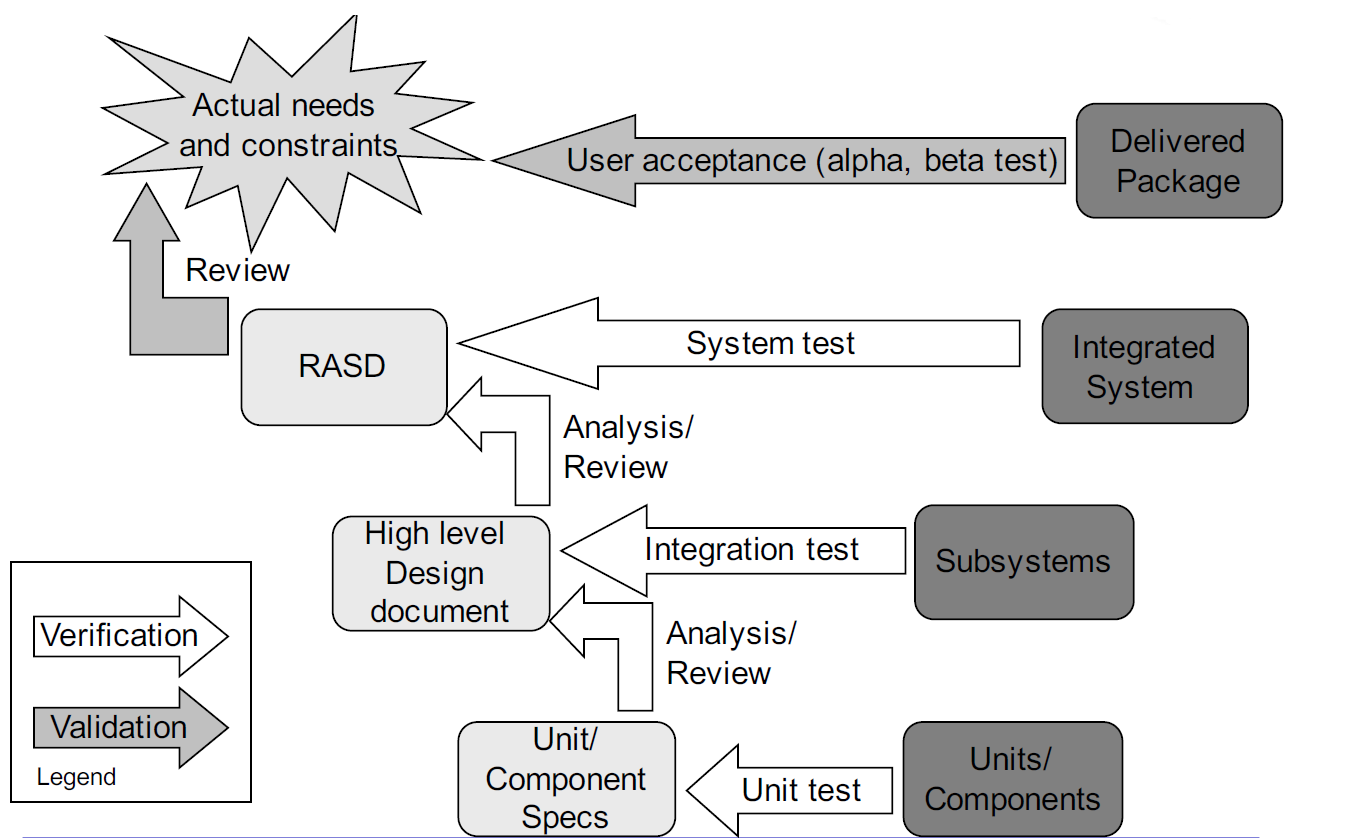
* ITPD: Integration Test Plan Document
* DD: Design document;
* ER: Entity-Relationship diagram;
* RASD: Requirements Analysis and Specification Document;
* JSE: Java Serial Edition;
* JEB: Java Enterprise Bean;
* REST: REpresantional State Transfer;
* RESTful service: REST compliant service;
* UX: User eXperience;
* JEE: Java Enterprise Edition;
* JAX-RS: Java API for RESTful web Services;
* JBOSS: JEE open source application server;
* JPA: Java Persistence API;
* Cordova: mobile cross-platform development framework;
* PhoneGap: mobile cross-platform development framework that works over Cordova;
* JSP: Java Server Pages.

## List of Reference Documents

* Specification Document: Assignments AA 2016-2017.pdf;
* RASD v1.1 Document;
* Design Document v1.0;
* Example Design Documents from previous years;

# Integration Strategy

## Entry Criteria



We are to design and plan the Integration Test, that aims to verify that software component work with each other and cooperate in the right and expected way.

Hence, it is supposed that each component works well individually and this can be formally proved with Unit Tests.

We assume that Low-level code is already tested, functions of every component are covered with unit tests, with mainly a white-box approach.

## Elements to be integrated

Starting from a high level view, it is necessary to integrate and test software tiers in the main server: clients with the web tier, web tier with the business logic tier, and this one with the persistence manager and the database. It is also necessary to test interaction between external handler and corresponding controller.

In a more low-level view, it is necessary to integrate every controller with related ones for example the Search Controller with the Map Controller.

It is also necessary to test the interaction between controllers developed in the car software, and between cars and server.

Since the client side is light, it is only necessary to test the interaction with clients and server.

A more detailed list of components to be integrated and tested is present further in the document.

## Integration Testing Strategy

For testing we choose the bottom-up approach. Since there wasn’t an old system, the project will be built up starting from the ground up.

By choosing bottom-up approach, it is possible to test integration of components as they are ready, with no further delay.

## Sequence of Component – Function Integration

### Software Integration Sequence

Components have to start to be integrated starting from low-level ones.

This process brings to different higher-level and integrated sub-systems.

#### Business Logic Components Integration

Thanks to the bottom-up approach, integration testing can be parallelized.

This is a set of integration test that can be performed in parallel, hence the order is not mandatory. They can be carried out as components are finished to be developed and unit tested.

#### Car Components Integration

Software components that run on-board the car are centred on the main controller, called Car Controller. In order to test integration, the Car Controller has to be finished. When the other controllers are finished, they can be integrated and tested with the main one.

### Subsystem Integration Sequence

After having integrated and tested components in the main server and in the car, it is possible to move on higher level software modules. It is possible to test the integration between tiers on the main server, client and server, server and database, and between car and server.

After doing this, it will be possible to test the whole system.

# Individual Steps and Test Description

# Tools and Test Equipment Required

# Program Stubs and Test Data Required

# Effort Spent