



PowerEnJoy
Software Engineering II

Integration Test Plan Document

Giovanni Scotti, Marco Trabucchi

Document version: 1
December 28, 2016

Contents

Contents	1
1 Introduction	2
1.1 Purpose and Scope	2
1.1.1 Purpose	2
1.1.2 Scope	2
1.2 Definitions, Acronyms, Abbreviations	2
1.3 Reference Documents	3
A Appendix	4
Bibliography	5

Section 1

Introduction

1.1 Purpose and Scope

1.1.1 Purpose

The Integration Test Plan Document (ITPD) is intended to provide the guidelines to accomplish the integration test phase planning in sufficient detail. This also includes determining which tools are needed and will be used during the testing process itself, as well as the required stubs, drivers and data structures that will be useful during said process.

1.1.2 Scope

PowerEnJoy is a car sharing service that only employs electric vehicles; it is provided for a large city, and aims to support the sharing process and car management of the electric cars, as well as the booking and payments for the service itself.

1.2 Definitions, Acronyms, Abbreviations

RASD: Requirements and Specification Document.

DD: Design Document.

ITPD: Integration Test Plan Document.

1.3 Reference Documents

The indications provided in this document are based on the ones stated in the previous deliverables for the project, the RASD document [1] and the DD document [2].

Moreover it is strictly based on the test plan example [?] presented during lectures and on the specifications concerning the RASD assignment [3] for the Software Engineering II project, part of the course held by professors Luca Mottola and Elisabetta Di Nitto at the Politecnico di Milano, A.Y. 2016/17.

Section 2

Integration Strategy

2.1 Entry Criteria

This section expresses the prerequisites needed to be met before the integration phase takes place.

Documentation: The documentation for every method and class must be provided for each individual component, in order to make it easier to reuse classes and understand their functioning; this is in fact also a prerequisite for the unit tests to be performed before the integration test phase. When necessary, a formal language specification of the classes' behaviours can be used (such as JML - Java Modelling Language).

Unit tests: All the classes and methods must be tested thoroughly using JUnit, in order to assure a properly correct behaviour of the internal mechanics of the individual components. It is required that the test coverage of each class and package reaches 90% of the code lines; moreover, test cases must be written with continuity and executed at every consecutive build of the project: this is needed in order to ensure that newly added lines do not interfere with the stability of the rest of the code.

Code Inspection and Analysis: Both automated data-flow analysis and code inspection must be performed on the whole project classes. This will reduce the risk that, during the integration test phase, any code-related issues or bugs rise, leading to more complex problematic situations to be solved in latter phases of the project development, with much greater effort for the development team.

RASD and DD: Along with the indications provided in this very document (ITPD), the two previous documents for this project, RASD and DD, must be delivered before the integration test phase can begin.

2.2 Elements to be Integrated

2.3 Integration Testing Strategy

2.4 Sequence of Components/Function Integration

Section 3

Individual Steps and Test Description

Section 4

Tools and Test Equipment Required

Section 5

Program Stubs and Test Data Required

Appendix A

Appendix

A.1 Software and tools used

- L^AT_EX, used as typesetting system to build this document.
- draw.io - <https://www.draw.io> - used to draw diagrams and mock-ups.
- GitHub - <https://github.com> - used to manage the different versions of the document and to make the distributed work much easier.
- GitHub Desktop, the GitHub official application that offers a seamless way to contribute to projects.

A.2 Hours of work

The absolute major part of the document was produced in group work. The approximate number of hours of work for each member of the group is the following:

- Giovanni Scotti:
- Marco Trabucchi:

NOTE: indicated hours include the time spent in group work.

Bibliography

- [1] AA 2016/2017 Software Engineering 2 - *Requirements Analysis and Specification Document* - Giovanni Scotti, Marco Trabucchi
- [2] AA 2016/2017 Software Engineering 2 - *Design Document* - Giovanni Scotti, Marco Trabucchi
- [3] AA 2016/2017 Software Engineering 2 - *Project goal, schedule and rules*
- [4] SpinGrid Project - *Integration Test Plan*