POLITECNICO
MILANO 1863

POWER ENJOY

Integration Test Plan Document

Lorenzo Casalino - 877421

Tommaso Castagna

Document version 1.0

Contents

1 Introduction

1.1 Revision History

The history of document revisions is here recorded in tabular format, mapping the document version with the major changes brought to.

The current version of the document is highlighted by the version number in bold format.

Version	Revision
1.0	Document first final version.

1.2 Purpose and Scope

1.2.1 Purpose

The *Integration Test Plan Document*, also referred to with the acronym of *ITDP*, aims to provide to the development team the path to follow for the integration testing process of the software system through a complete description of the elements of the system to test, the integration strategy to adopt, the integration sequence forecasted and stubs/drivers and tools needed to accomplish the integration test phase.

1.2.2 Scope

The hereby Integration Test Plan Document presents a detailed description of the integration testing plan that the development team should follow to accomplish correctly the integration testing process.

This document contains a detailed description of the integration testing plan that the development team should follow to successfully complete the integration testing process.

The scope of this document covers four main topics regarding the integration phase. Here, the scope is presented reflecting the main structure of the ITPD, allowing the reader to better understand what is discussed in the following sections.

- **Integration Strategy:** the entry criteria, both general and specific, that must be met before the integration test take place, what are the subsystems to integrate, which integration strategies are adopted and the integration sequence to follow are deeply described and discussed, pointing out the rationale of each choice.

- **Individual Steps and Test Description:** the type of test performed in each step of the integration process and applied to each component and subsystem, along with the expected results, are presented.
- **Tools and Test Equipment:** the needed tools to perform the integration are presented, and briefly described their usage during the integration process.
- **Program Stubs and Test Data Required:** possible stubs/drivers or special data required to proceed in each step of the integration process are presented and described.

1.3 Definitions, Acronyms and Abbreviations

1.3.1 Definitions

- **Software Component:** atomic piece of software that compose the software system.
- **Design Document**: document describing the software system architecture, the design choices and their rationale.
- **Integration Test Phase:** project development phase where the components and subsystems of the software system designed are integrated and tested in order to verify the correctness of their implementation and presence of design flaws.
- **Integration Test Plan Document:** document used to guide the integration test phase.
- **Requirement Analysis and Specification Document:** document regarding the analysis of the goals of the project stakeholder and the functional and non-functional requirements of the software system to develop.
- **Software System:** the software system that is currently under development.
- **Subsystem:** part of the software system composed by two ore more software components.

1.3.2 Acronyms

- **DD**: Design Document.
- **ITPD**: Integration Test Plan Document.
- **RASD**: Requirement Analysis and Specification Document.

1.3.3 Abbreviations

- **Integration phase:** integration test phase
- **Integration process:** integration test process
- **System:** software system.

2 Integration Strategy

2.1 Entry Criteria

Before the integration testing phase of specific components may take place, it is fundamental that the following criteria (or conditions) here described are satisfied.

The verification of these conditions is really important in order to have as outputs of the integration test phase meaningful results, useful to assess the quality of the software system designed and, possibly, improve it.

It's worthful point out that some of the criteria presented are strictly tied to the kind of strategy chosen to perform the integration testing of the system's components, while others are more general and act as pre-condition for the whole integration test process.

For informations about the integration testing strategy picked out for this software system, please refer to the *Integration Testing Strategy* section.

GENERAL CRITERIA

- **RASD complete draw up:** the RASD has the main function of thoroughly document the functionalities and requirements of the software system. Because of its purpose, it is the main source of informations and comparison that the development team has to refer to check the results obtained after the integration test of components and subsystems.
- **RASD positive assessment**: since the development team refers to the RASD to veirify the results of the integration tests, it's fundamental that the RASD content reflects the goals of the stakeholders involved into the project. Documented functionalities diverging from the stakeholders' desires lead to wrong implementations, regardless of the integration test results.
- **DD complete draw up:** the design document has the purpose to describe the system's architecture and the design choices taken. These informations are essential in the development phase, where the implementation of the described software architecture take place. Furthermore, the verification of the result obtained from the integration test of subsystems relies on the content described in the DD.
- **DD positive assessment:** development and integration phase rely heavily on the informations contained in the Design Document, therefore the positive assessment of its content is an essential criterion for the correct development and integration of system's components.
- **ITPD complete draw up:** the ITPD describes all the aspects regarding the integration test phase. Without it, the integration process cannot take

place.

- **ITPD positive assessment:** the integration test choices and the path planned must comply and being consistent with the RASD and DD documents. A divergence between what is stated in the ITPD document and what is stated in the RASD and/or in the DD compromises the entire integration and development phases.

SUBSYSTEMS AND COMPONENTS TEST CRITERIA

- **Creation of required stubs:** some components may need the support of functionalities provided by components not already developed. To overcome this obstacle, the scaffolding process comes in play, designing the stubs needed to emulate the required components.
- **Complete component static and dynamic analysis:** the supporting functionalities provided by the components to be integrated must thoroughly inspected through static analysis methods, such as code inspection, and dynamic analysis methods, i.e. unit testing. This step is important because allows to discover possible fault in the implementation, easing and speeding up the integration testing process.
- **Regression testing:** before starting the integration testing of a certain functionality through new test cases, the old test cases should be run in order to verify the compatibility and the correctenes of the new integration. This process should be executed before the new functionality testing because if the old tests show an incompatibility, the components integration must be reviewed.

2.2 Elements to be Integrated

In the following, the system's elements to be integrated are spotted and described. With *elements* here we mean the subsystems composing the system's design. The description of a subsystem is recursive, meaning that if a subsystem is composed by other subsystems, even these are described. Description of atomic components is avoided since are well described into the DD.

The main *elements* composing our system are:

- **Data Tier:** This element represents the DBMS. Even though we are not developing the DBMS itself, it is still part of the system, hence it has to be integrated.
- **Business Logic:** This element contains all the application logic present in our system. The main components of this element are: the **Authentication Manager**, the **Account Manager**, the **Maintenance Manager**, the **Vehicle Manager** and the **Reservation Manager**.

- **Web Server:** This is the element responsible for the communications between the business logic tier and the clients. It exposes a RESTful implementation of the communication interfaces.
- **Client:** This element contains the three different kind of clients that can access our application, namely the **mobile client**, the **web application client** and the **car central system**

Every component that forms the different *elements* has aledy been tested individually, the tests to be performed are applied to the interfaces that connect a component with another one.

2.3 Integration Testing Strategy

The system's architecture design, presented in [DD 2.2] and described in more detail in [DD 2.3], shown a relatively low architecture complexity. Functionality-oriented strategies, i.e. thread strategy, or critical-modules-oriented strategies are not considered appropiated since the planning efforts needed exceed the possible outcomes derived from the use of these strategies.

Structural-oriented strategies, instead, shown to be quite simply to plan and apply. Among the possible strategies of this testing category, the one considered most indicated to approach the integration testing activity is the **Top-Down Strategy**.

The choice is driven by three main features that characterize this testing approach:

- **Unit test prioritization:** after a component is implemented, it must be immediately tested before any other activity, i.e. implementation or testing another component, can be performed. This means that possible implementation errors, like functionality discrepancies or performance issues, may be discovered immediately after the implementation phase.
- **Ease design flaws detection:** structural errors in the software system's architecture can be easier detected, avoiding extensive design re-implementation or correction.
- **Early availability of a prototype:** the presence of a prototype allows an early validation stage. This means that eventual discrepancies between what has been designed in terms of functionalities and what has been developed may be detected in due time.

3 Effort Spent

The effort spent by each member of the group in terms of hours is shown in the following:

Lorenzo Casalino

- 27 December 2016 - 1h 10m
- 28 December 2016 - 1h
- 29 December 2016 - 1h 50m
- 30 December 2016 - 1h
- 02 January 2017 - 1h

Tommaso Castagna

- 29/12/16 - 1h30m