Politecnico di Milano
A.A. 2015-2016
Software Engineering 2: “myTaxiService”

Integration **T**est **P**lan **D**ocument

Roberto Clapis (841859), Erica Stella (854443)

January 20, 2016

Contents

1 Introduction

1.1 Purpose and Scope

This document describes the Integration Test Plan for the myTaxiService application. Its purpose is to provide a plan for how the application's components, listed in the Design Document, will be integrated for testing. It also provides a description of the necessary tools, stubs and drivers.

1.2 List of Definitions and Abbreviations

- Abbreviations:
  - *UI:* User Interface.
- Definitions:
  - *Database's stubs:* Active Requests and Reservations' stub and Accounts' stub.
  - *UIs' stubs:* ClientUI's stub and DriverUI's stub.

1.3 List of Reference Documents

- The document with myTaxiService's description
- myTaxiService's RASD
- myTaxiService's Design Document

2 Integration Strategy

2.1 Entry Criteria

Before the integration testing, each single module must have been tested to verify the correct functioning of its methods according to their specifications.

2.2 Elements to be Integrated

According to the Design Document, the components to be integrated are:

- Database:
  - Accounts
  - Active Reservations and Requests
- Web Server:
  - DBConnector

- APIBackend
- WebpageCreator
- NotificationModule
- HttpHandler

- UI:
  - ClientUI
  - DriverUI
  - AdminUI

2.3 Integration Testing Strategy

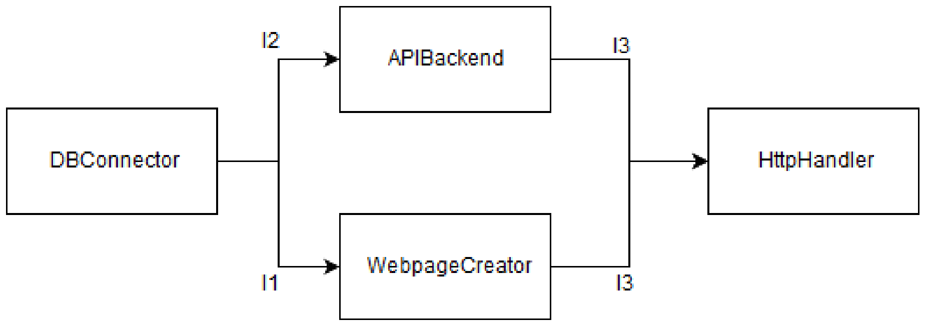
The decided testing approach is sandwich. It has been chosen in order to integrate first the modules of the WebServer, and then integrate it with the Database and the UI. This is because the WebServer is the only application part made by connected subcomponents, and is also its core. Adopting this approach made possible to test thoroughly the correct functioning of the WebServer with controlled input and without interferences from other parts of the application.

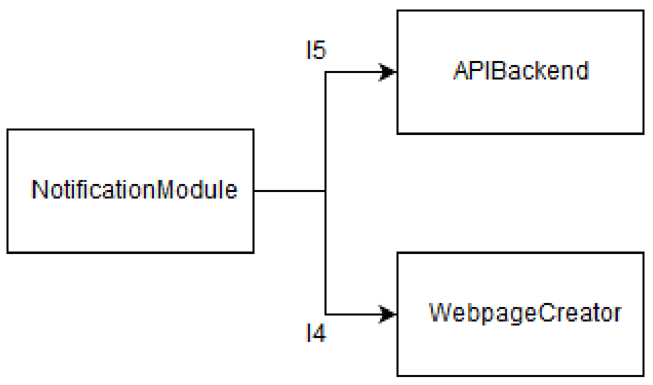
2.4 Sequence of Component/Function Integration

2.4.1 Software Integration Sequence

In the following graphs the arrows go from the called module to the caller module and are marked with identifiers that define the order of integration.

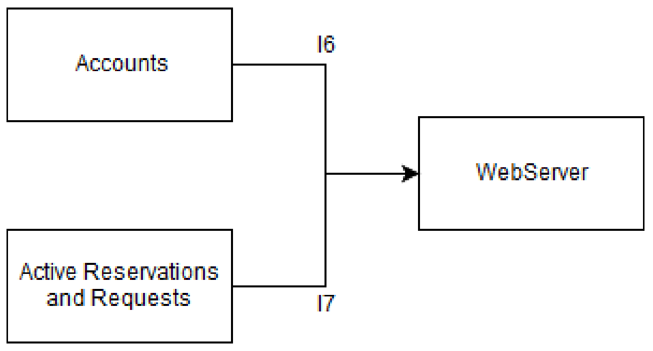
2.4.1.1 Web Server The following images describe how the Web Server's components will be integrated for testing.

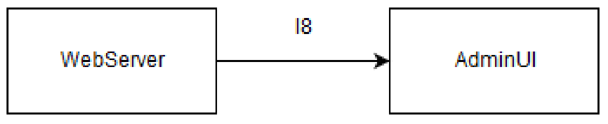


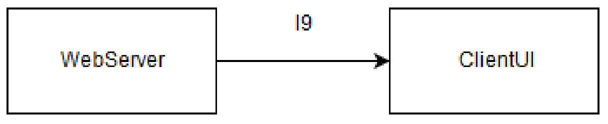


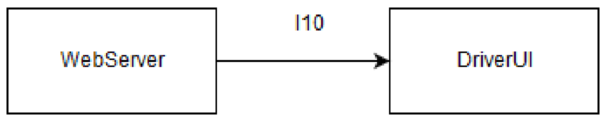
2.4.2 Subsystem Integration Sequence

The following graphs describe how the three main components of myTaxiService application, the Database, the Web Server and the UIs, will be integrated.









3 Individual Steps and Test Description

3.1 Test case specifications

3.1.1 I1

Test Case identifier	I1T1
Test Items	DBConnector → WebpageCreator
Input Specification	Perform valid and invalid requests on the WebpageCreator
Output Specification	All and only the queries that should be allowed are executed and the correct DBConnector's methods are called
Environmental Needs	Database's stubs, WebpageCreator driver

3.1.2 I2

Test Case identifier	I2T1
Test Items	DBConnector → APIBackend
Input Specification	Perform valid and invalid requests on the APIBackend
Output Specification	All and only the queries that should be allowed are executed and the correct DBConnector's methods are called
Environmental Needs	Database's stubs, APIBackend driver

3.1.3 I3

Test Case identifier	I3T1
Test Items	WebpageCreator → HttpHandler
Input Specification	Perform valid and invalid requests on the HttpHandler
Output Specification	Verify if only the requests intended for the WebpageCreator are forwarded to it and the correct WebpageCreator methods are called
Environmental Needs	Database's stubs, HttpHandler driver, I1,I2 successful
Test Case identifier	I3T2
Test Items	APIBackend → HttpHandler
Input Specification	Perform valid and invalid requests on the HttpHandler
Output Specification	Verify if only the requests intended for the APIBackend are forwarded to it and the correct APIBackend methods are called
Environmental Needs	Database's stubs, HttpHandler driver, I1,I2 successful

3.1.4 I4

Test Case identifier	I4T1
Test Items	NotificationModule → WebpageCreator
Input Specification	All the possible types of input that require sending notifications to a client
Output Specification	Check if the correct methods of the NotificationModule have been called
Environmental Needs	UIs' stub, WebpageCreator driver

3.1.5 I5

Test Case identifier	I5T1
Test Items	NotificationModule → APIBackend
Input Specification	All the possible types of input that require sending notifications to a client
Output Specification	Check if the correct methods of the NotificationModule have been called
Environmental Needs	UIs' stub, APIBackend driver

3.1.6 I6

Test Case identifier	I6T1
Test Items	Accounts → WebServer
Input Specification	Queries to manipulate (creation, modification and deletion) of accounts
Output Specification	The correct Accounts' method have been called
Environmental Needs	WebServer driver, Active Reservations and Requests' stub, I1-15 successful

3.1.7 I7

Test Case identifier	I7T1
Test Items	Active Reservations and Requests → WebServer
Input Specification	Queries to place/delete reservations and requests, in every possible order of execution
Output Specification	The correct Active Reservations and Requests' methods have been called
Environmental Needs	WebServer driver, I1-I6 successful

3.1.8 I8

Test Case identifier	I8T1
Test Items	WebServer → AdminUI
Input Specification	Every possible input from the UI
Output Specification	The correct WebServer's methods have been called
Environmental Needs	I1-I7 successful

3.1.9 I9

Test Case identifier	I9T1
Test Items	WebServer $\rightarrow$ ClientUI
Input Specification	Every possible input from the UI
Output Specification	The correct WebServer's methods have been called
Environmental Needs	I1-I7 successful

3.1.10 I10

Test Case identifier	I10T1
Test Items	WebServer $\rightarrow$ AdminUI
Input Specification	Every possible input from the UI
Output Specification	The correct WebServer's methods have been called
Environmental Needs	I1-I7 successful

3.2 Integration Test Procedures

3.2.1 TP1

Test Procedure Identifier	TP1
Purpose	This test procedure verifies whether the WebServer's components work properly together with provided input and aside from the database
Procedure Steps	Execute I1-I5

3.2.2 TP2

Test Procedure Identifier	TP2
Purpose	This test procedure verifies whether the Database can handle all types of inputs and modifications requested by the WebServer
Procedure Steps	Execute I6-I7

3.2.3 TP3

Test Procedure Identifier	TP3
Purpose	This test procedure verifies whether the WebServer can handle all the inputs from the UIs and outputs the requested information
Procedure Steps	Execute I8-I10

4 Tools and Test Equipment Required

We base our tools' choice on the assumption that the implementation of the myTaxiService application has been made using Java as programming language.

As one of the most used and reliable testing frameworks currently available, we decided to exploit the functionalities of Mockito. In particular, it has been used to create all the stubs and the drivers named in Section 3.1.

5 Program Stubs and Test Data Required

5.1 Stubs

The required stubs are:

- Active Requests and Reservations' stub
- Accounts' stub
- ClientUI's stub
- DriverUI's stub

For the complete set of methods exposed by the interfaces of the modules listed above, refer to the Design Document's section 2.6.

5.1.1 Database' stub

This stubs should simulate a real database. They should provide responses for all the possible contents a database could have.

5.1.2 UIs's stub

These stubs are used only in the NotificationModule's test. They have to mock the reception of messages from the WebServer.

5.2 Drivers

As stated in Section 3.1, all tests except for I8-I10 need a driver. They are used to provide the inputs that should cover all the possible inputs, where possible, the module could receive and check the output for correctness.