

Politecnico di Milano

A.A. 2015-2016

Software Engineering 2 Project

**myTaxiService**

**Integration Test Plan Document**

**Ver. 1**

**January 21, 2016**

Alessio Martorana - 860584

[1 Introduction 3](#_Toc441007173)

[1.1 Revision History 3](#_Toc441007174)

[1.2 Purpose and Scope 3](#_Toc441007175)

[1.3 List of Definitions and Abbreviations 3](#_Toc441007176)

[1.4 List of Reference Documents 3](#_Toc441007177)

[2. Integration Strategy 4](#_Toc441007178)

[2.1 Entry Criteria 4](#_Toc441007179)

[2.2 Elements to be integrated 4](#_Toc441007180)

[2.3 Integration Testing Strategy 5](#_Toc441007181)

[2.4 Sequence of Component Integration 6](#_Toc441007182)

[2.4.1 Software Integration Sequence 7](#_Toc441007183)

[2.4.1 Subsystem Integration Sequence 13](#_Toc441007184)

[3. Individual Steps and Test Description 14](#_Toc441007185)

[3.1 Authentication subsystem test procedure 14](#_Toc441007186)

[3.1.1 Integration test case I1 14](#_Toc441007187)

# 1 Introduction

## 1.1 Revision History

No revisions of the document at the moment.

## 1.2 Purpose and Scope

This document describes the test plan for components integration.

The purpose of the document is to test module interactions and interfaces with which components described in DD interacts. This document is brought to the attention to every person involved in integration test phase.

The system implements a taxi driver rides handling, its main function is to permit the exchange of necessary data to handle request taxi rides between users and taxi drivers, users can interface to the system either with a web application or a mobile app, taxi drivers can serve request through a specific mobile app.

## 1.3 List of Definitions and Abbreviations

* **myTaxiService:** The name of the system developed
* **RASD:** Requirement Analysis and Specification Document
* **DD:** Design Document

## 1.4 List of Reference Documents

* **Assignment 1,2,3 and 4:** Assignments for the various phase of the project given to me by the professor.
* **RASD:** The Requirement Analysis and Specification Document
* **DD:** The Design Document
* **THE DOCUMENTATION OF ANY TOOL I PLAN TO USE FOR TESTING**

# 2. Integration Strategy

## 2.1 Entry Criteria

Before integration testing may begin, the following documents must be released:

* TheDesign Document (DD)
* The Requirement Analysis and Specification Document (RASD)
* The Integration Test Plan Document (ITPD), that is this document

Before integration testing may begin, is also needed that:

* The single components involved have been delivered
* Involved single components have been unit tested and unit testing test reports concerning these components have been delivered
* Driver for the integration test have been delivered
* Input data for the integration test has been delivered

## 2.2 Elements to be integrated

As described in the DD, the component view of myTaxiService system is the one showed in the following component diagram: [DA TOGLIERE??]

Referring to the DD, the components to be integrated and integration tested are the following:

* **myTaxiService web application**
* **myTaxiService users mobile app**
* **myTaxiService taxis mobile app**
* **Controller**
* **Authenticator**
* **Authentication checker**
* **Logged clients information handler**
* **Dispatcher**
* **Ride request handler**
* **Taxi driver handler**

## 2.3 Integration Testing Strategy

The integration testing approach chosen is the “Bottom-up” one, because due to the non-excessive subsystems dimension and the modularity architecture of the system, this strategy is considered as a good approach.

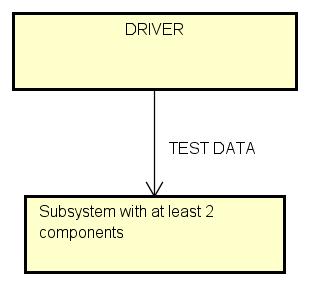
The “Bottom-up” integration testing strategy starts from the bottom level of the system, trying to test the integration of the leaves of the “uses” hierarchy, and going up and up to the top level of the system as lowest levels become already tested.

“Bottom-up” integration testing strategy needs some additional components called “drivers”, which are to interface with each subsystems module, passing the test data to the component of the subsystem to be tested and printing results.

## 2.4 Sequence of Component Integration

Here the components and subsystems integration sequence is described, at first a diagram is presented, in which the arrows have two meanings: the arrows with an ID (ex. I1) between brackets represent the integration sequence of the objects, in the sense that if object A is placed before the arrow and object B is placed after the arrow, object B will be integrated after object A and the ID represent the identifier of the integration test that has to be done, the arrows with “TEST DATA” instead represent the calls of the modules by a specific driver for each subsystem identified.

As the integration testing strategy is bottom up, every step of components integration in subsystems is tested with a driver as shown in the following general scheme



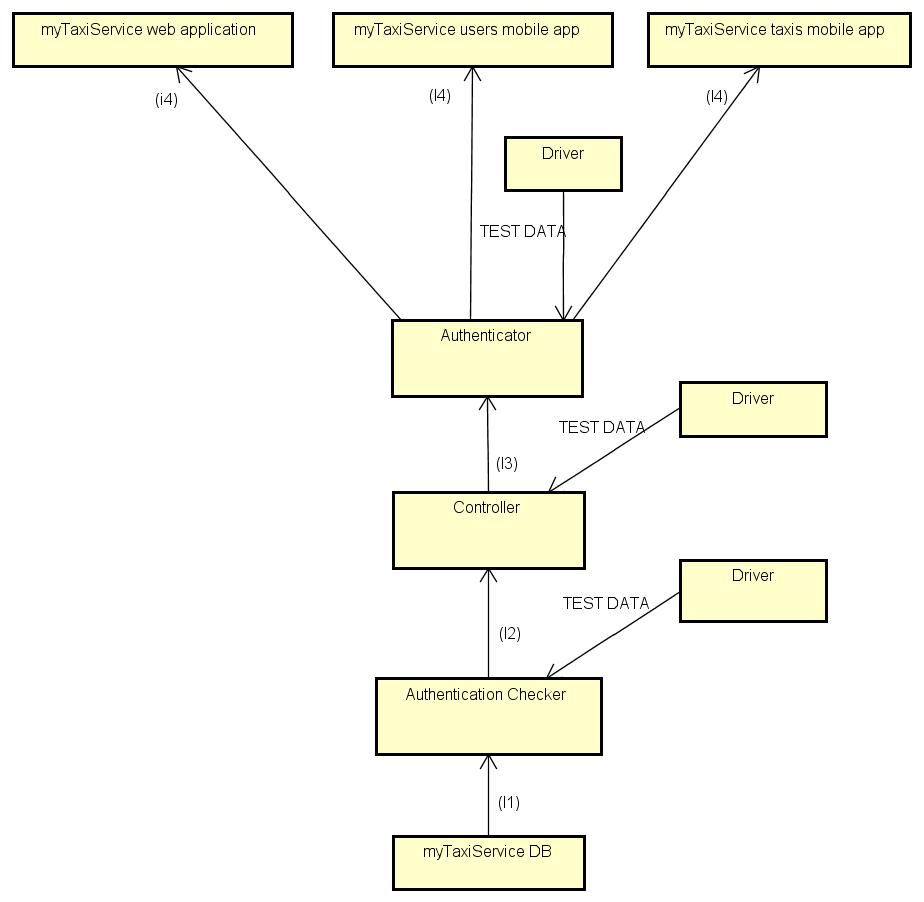
From the figure can be seen that, as the components are supposed to be already unit tested, the subsystem tested is considered to be at least of two components.

Under each diagram a table with the details on components integration is presented, ordered by ID: in this table the arrow represents only the order of interactions of an object by another one, so – for example - if object A is placed before the arrow and object B is placed after the arrow, object A will interact with object B calling one of its methods/functions; as can be seen, calls by driver are notified; if more than one object interacts with the object after the arrow, the objects before the arrow are listed separated by comma. The column “paragraph” indicates the paragraph in this document where the description of the type of tests for the integration are stated.

### 2.4.1 Software Integration Sequence

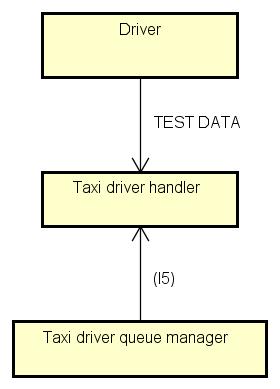
In this paragraph the software components integration sequence is described for each subsystem.

**Authentication subsystem**

****

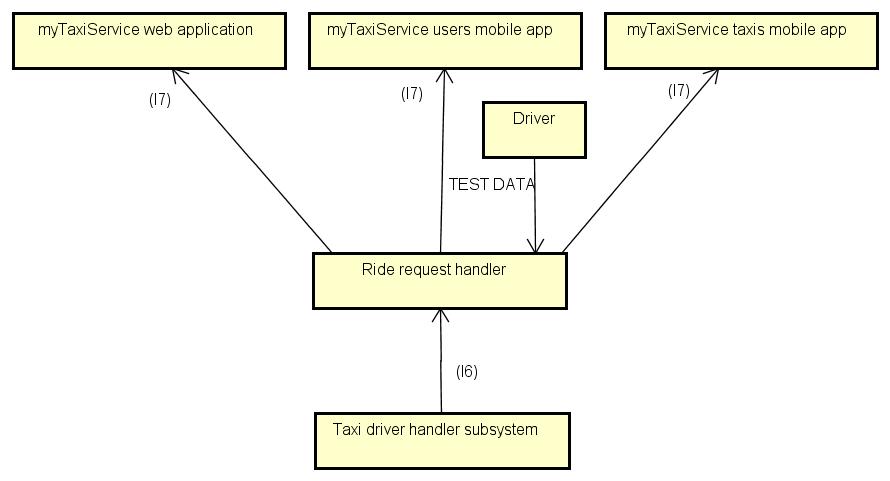
|  |  |  |
| --- | --- | --- |
| **ID** | **Integration Test** | **Paragraph** |
| I1 | *Driver* → Authentication Checker → myTaxiService DB | 3.1.1 |
| I2 | *Driver* → Controller → Authentication Checker | 3.1.2 |
| I3 | *Driver* → Authenticator → Controller | 3.1.3 |
| I4 | MyTaxiService web application, MyTaxiService users mobile app, MyTaxiService taxis mobile app → Authentication Checker | 3.1.4 |

**Taxi driver handling subsystem**

****

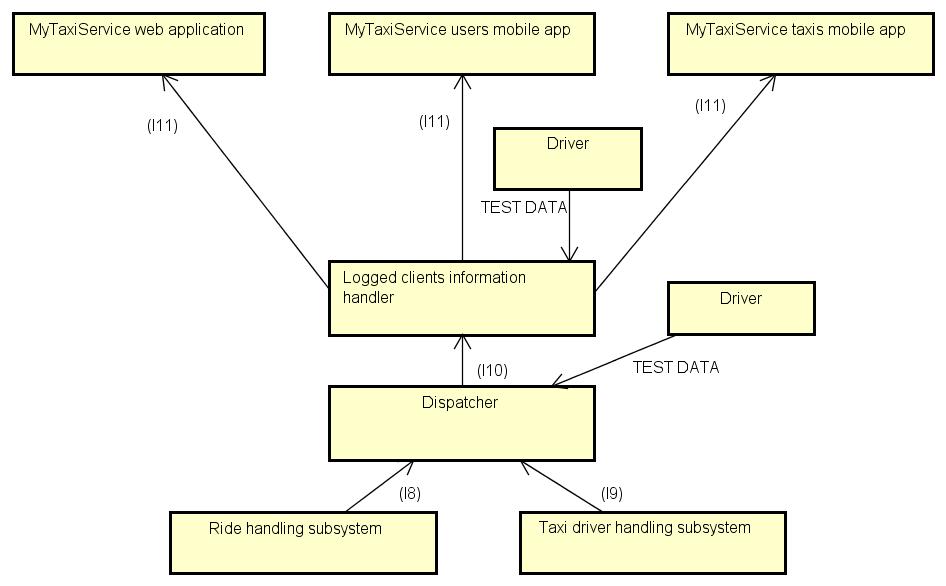
|  |  |  |
| --- | --- | --- |
| **ID** | **Integration Test** | **Paragraph** |
| I5 | *Driver*→ Taxi driver handler → Taxi driver queue manager | 3.2.1 |

**Ride handling subsystem**

****

|  |  |  |
| --- | --- | --- |
| **ID** | **Integration Test** | **Paragraph** |
| I6 | *Driver*→ Ride request handler → Taxi driver handler subsystem | 3.3.1 |
| I7 | MyTaxiService web application, MyTaxiService users mobile app, MyTaxiService taxis mobile app → Authentication Checker | 3.3.2 |

**Dispatching subsystem**

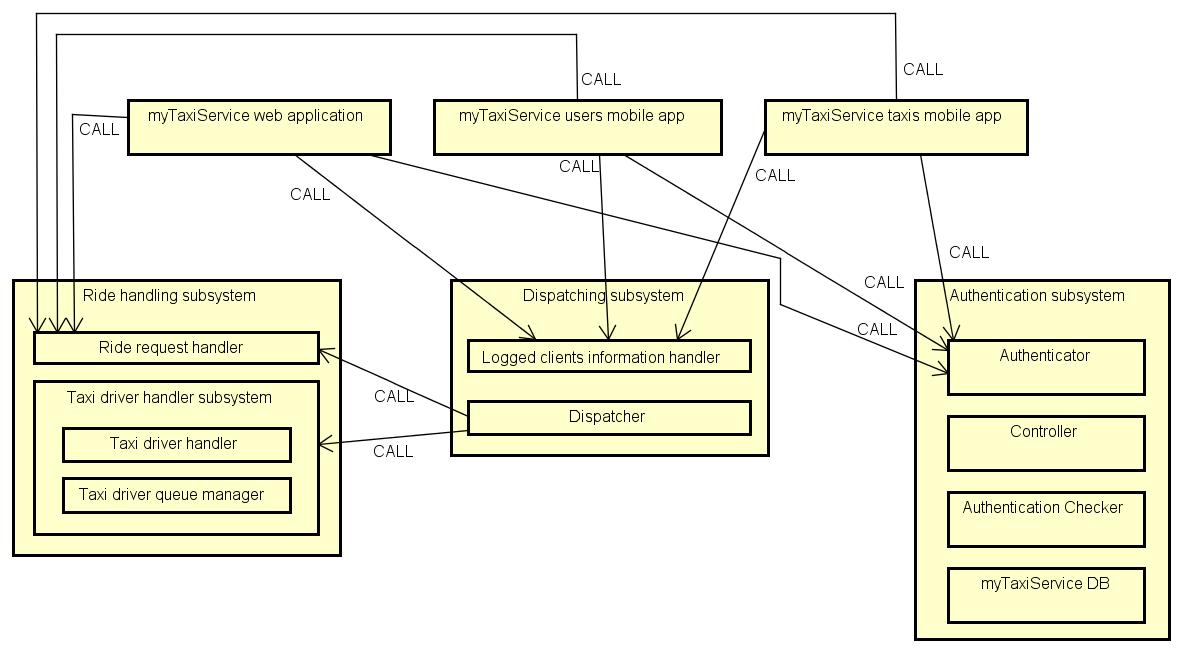
****

|  |  |  |
| --- | --- | --- |
| **ID** | **Integration Test** | **Paragraph** |
| I8 | *Driver*→ Dispatcher → Ride request handler subsystem | 3.4.1 |
| I9 | *Driver*→ Dispatcher → Taxi driver handling subsystem | 3.4.2 |
| I10 | *Driver*→ Logged clients information handler → Dispatcher | 3.4.3 |
| I11 | MyTaxiService web application, MyTaxiService users mobile app, MyTaxiService taxis mobile app → Ride request handler | 3.4.4 |

**asda**

## 2.4.1 Subsystem Integration Sequence

As the creation of subsystems shown above proceeded using smaller subsystems to create bigger ones, the subsystem integration sequence simply consists in the integration of all the subsystems together through the “interface components” - that are the components by which the subsystems interface each other – at the same time, all the once, to get the entire system and test it by the client applications (web application, users mobile app, taxis mobile app), as shown in the diagram that follows, where the arrows represents the interaction (calls of functions) between the components, with the same meaning of the arrows in the previous tables.



# 3. Individual Steps and Test Description

In this section of the document the type of tests that will be used in the integration test will be described. For each subsystem the test procedure is reported and the integration steps are identified, also, for each specific step of the integration, relative sub-steps are described.

## 3.1 Authentication subsystem test procedure

|  |  |
| --- | --- |
| **Test procedure identifier** | AS-TP |
| **Purpose** | This test procedure verifies whether the Authentication subsystem:   * Correctly handles clients input * Correctly stores user information in the database * Correctly checks log-in information * Correctly communicates registration operation result to clients * Correctly communicates log-in operation result to clients |
| **Procedure steps** | Execute I1,I2,I3 and I4 in this precise order |

### 3.1.1 Integration test case I1

|  |  |
| --- | --- |
| **Test case identifier** | I1 |
| **Test items** | *Driver* → Authentication Checker → myTaxiService DB |
| **Input specification** | Create a typical Authentication Checker input |
| **Output specification** | Check if the correct method/functions are called in myTaxiService DB |
| **Environmental needs** | Authentication checker driver |

### 3.1.2 Integration test case I2

|  |  |
| --- | --- |
| **Test case identifier** | I2 |
| **Test items** | *Driver* → Controller → Authentication Checker |
| **Input specification** | Create a typical Controller input |
| **Output specification** | Check if the correct method/functions are called in Authentication Checker |
| **Environmental needs** | I1 succeded, Controller driver |

### 3.1.3 Integration test case I3

|  |  |
| --- | --- |
| **Test case identifier** | I3 |
| **Test items** | *Driver* → Authenticator → Controller |
| **Input specification** | Create a typical Authenticator input |
| **Output specification** | Check if the correct method/functions are called in Controller |
| **Environmental needs** | I2 succeded, Authenticator driver |

### 3.1.4 Integration test case I4

|  |  |
| --- | --- |
| **Test case identifier** | I4 |
| **Test items** | MyTaxiService web application, MyTaxiService users mobile app, MyTaxiService taxis mobile app → Authentication Checker |
| **Input specification** | Create a typical inputs for MyTaxiService web application, MyTaxiService users mobile app and MyTaxiService taxis mobile app |
| **Output specification** | Check if the correct method/functions are called in Authentication Checker |
| **Environmental needs** | I3 succeded |

## 3.2 Taxi driver handling subsystem test procedure

|  |  |
| --- | --- |
| **Test procedure identifier** | TDHS-TP |
| **Purpose** | This test procedure verifies whether the Taxi driver handling subsystem:   * Correctly handles taxi driver’s position input * Correctly handles Ride requests inputs * Correctly outputs taxi driver’s data (ID, position, availability etc…) * Correctly (fairly) manages taxi drivers queue |
| **Procedure steps** | Execute I5 |

### 3.2.1 Integration test case I5

|  |  |
| --- | --- |
| **Test case identifier** | I5 |
| **Test items** | *Driver* → Authenticator → Controller |
| **Input specification** | Create a typical Taxi driver handler input |
| **Output specification** | Check if the correct method/functions are called in Taxi driver queue manager |
| **Environmental needs** | Taxi driver handler driver |

## 3.3 Ride handling subsystem test procedure

|  |  |
| --- | --- |
| **Test procedure identifier** | RHS-TP |
| **Purpose** | This test procedure verifies whether the Rie handling subsystem:   * Correctly handles clients input * Correctly handles ride requests client inputs * Correctly outputs ride request results to clients |
| **Procedure steps** | Execute I6 and I7 in this precise order |

### 3.3.1 Integration test case I6

|  |  |
| --- | --- |
| **Test case identifier** | I6 |
| **Test items** | *Driver*→ Ride request handler → Taxi driver handler subsystem |
| **Input specification** | Create a typical inputs for Ride request handler |
| **Output specification** | Check if the correct method/functions are called in Taxi driver handler subsystem |
| **Environmental needs** | Ride request handler driver |

### 3.3.2 Integration test case I6

|  |  |
| --- | --- |
| **Test case identifier** | I7 |
| **Test items** | MyTaxiService web application, MyTaxiService users mobile app, MyTaxiService taxis mobile app → Authentication Checker |
| **Input specification** | Create a typical inputs for MyTaxiService web application, MyTaxiService users mobile app and MyTaxiService taxis mobile app |
| **Output specification** | Check if the correct method/functions are called in Authentication checker |
| **Environmental needs** | I6 succeded |

DISPATCHING SUBSYSTEM E INTEGRAZIONE FINALE