POLITECNICO DI MILANO

Software Engineering 2 Project

myTaxiService

Integration Test Plan Document

Simone Guidi

Matteo Imberti

January 21, 2016

Contents

1 Introduction

1.1 Revision History

Version	Date	Authors	Summary
1.0	21/01/2016	Simone Guidi, Matteo Imberti	Document Creation

1.2 Purpose and Scope

This document describes the plans for testing the integration of the created components. The purpose of this document is to test the interfaces between the components as described in [SDD, Section 2]. Every team member who cooperates in the integration tests should read this document.

1.3 List of Reference Documents

- RASD.pdf
- SDD.pdf
- Assignments 1 and 2 (RASD and DD).pdf

2 Integration Strategy

2.1 Entry Criteria

Before the beginning of the integration test phase the following entry condition must be matched:

- Unit Tested Components/Modules
- All High prioritized bugs fixed and closed
- All Modules to be code completed and integrated successfully
- Integration test Plan, test case, scenarios to be signed off and documented
- Required Test Environment to be set up for Integration testing

2.2 Elements to be Integrated

2.2.1 Client Components

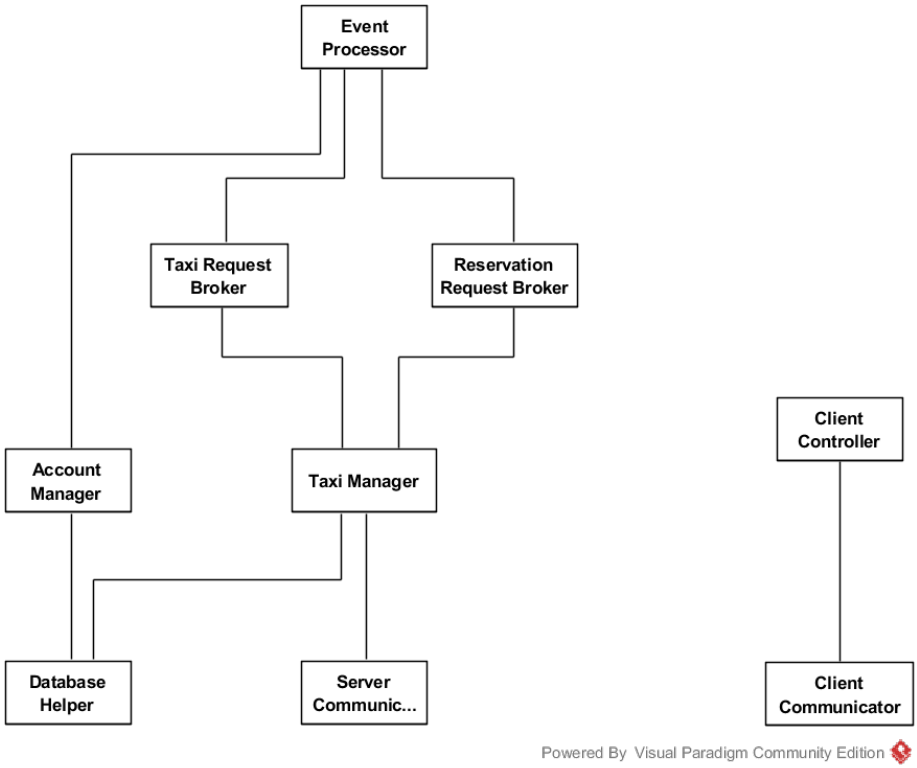
- Client Controller
- Client Communicator

2.2.2 Server Components

- Server Communicator
- Database Helper
- Account Manager
- Taxi Manager
- Reservation Broker
- Taxi Request Borker
- Event Processor

2.3 Integration Testing Strategy

We chose the bottom-up approach. This means that integration testing starts at the bottom level. This way the project will be built up from the bottom level. The integration tests described in this documents are at the component level.
We have identified different hierarchical levels of component on both the Client and the Server side. This hierarchy is illustrated in the diagram below.



2.4 Sequence of Component Integration

According to the bottom-up approach we adopted and the identified hierarchy, the sequence for the component integration is the following:

Client Side

1. Client Communicator - Client Controller

Server Side

1. Database Helper - Account Manager
2. Database Helper, Server Communicator - Taxi Manager
3. Taxi Manager - Taxi Request Broker
4. Taxi Manager - Reservation Request Broker
5. Account Manager, Taxi Request Broker, Reservation Request Broker - Event Processor

3 Individual Steps and Test Description

3.1 Integration test case I1

Test Case Identifier	I1T1
Test Item(s)	Client Communicator -> Client Controller
Input Specification	Create typical Client Communicator input
Output Specification	Check if the correct operations are executed in the Client Controller
Enviromental Needs	N/A

Test Case Identifier	I1T2
Test Item(s)	Client Controller -> Client Communicator
Input Specification	Create typical Client Controller input
Output Specification	Check if the correct operations are executed in the Client Communicator
Enviromental Needs	N/A

3.2 Integration test case I2

Test Case Identifier	I2T1
Test Item(s)	Account Manager -> Database Helper
Input Specification	Create typical Account Manager queries (e.g. login, registration,..)
Output Specification	Check if the query output is the expected result
Enviromental Needs	Database source (stub)

3.3 Integration test case I3

Test Case Identifier	I3T1
Test Item(s)	Taxi Manager -> Server Communicator
Input Specification	Create typical Taxi Manager input
Output Specification	Check if expected operations are executed by the Server Communicator
Enviromental Needs	N/A

Test Case Identifier	I3T2
Test Item(s)	Taxi Manager -> Database Helper
Input Specification	Create typical Taxi Manager queries
Output Specification	Check if the query output is the expected result
Enviromental Needs	Database source (stub)

3.4 Integration test case I4

Test Case Identifier	I4T1
Test Item(s)	Taxi Request Broker -> Taxi Manager
Input Specification	Create typical Taxi Request Broker input
Output Specification	Check if expected operations are executed by the Taxi Manager
Enviromental Needs	I3 succeded

3.5 Integration test case I5

Test Case Identifier	I5T1
Test Item(s)	Reservation Request Broker -> Taxi Manager
Input Specification	Create typical Reservation Request Broker input
Output Specification	Check if expected operations are executed by the Taxi Manager
Enviromental Needs	I3 succeded

3.6 Integration test case I6

Test Case Identifier	I6T1
Test Item(s)	Event Processor -> Account Manager
Input Specification	Create typical Event Processor input
Output Specification	Check if expected operations are executed by the Account Manager
Enviromental Needs	I2 succeded

Test Case Identifier	I6T2
Test Item(s)	Event Processor -> Taxi Request Broker
Input Specification	Create typical Event Processor input
Output Specification	Check if expected operations are executed by the Taxi Request Broker
Enviromental Needs	I4 succeded

Test Case Identifier	I6T3
Test Item(s)	Event Processor -> Reservation Request Broker
Input Specification	Create typical Event Processor input
Output Specification	Check if expected operations are executed by the Reservation Request Broker
Enviromental Needs	I5 succeded

Test Case Identifier	I6T4
Test Item(s)	Event Processor -> Server Communicator
Input Specification	Create typical Event Processor input
Output Specification	Check if expected operations are executed by the Server Communicator
Enviromental Needs	N/A

4 Tools and Test Equipment Required

Testing is done manually by direct observation of the behaviour of the involved subsystems and exploting the tools that an IDE such as Eclipse provides, for example debug mode and errors/exceptions logging.

5 Program Stubs and Test Data Required

In order to test the integration of the components a stub for the database is required. It should contain some profiles, at least one for each kind of user (passenger, taxi driver, etc), and some reservation request data.
There is no need of other stubs or drivers since all the modules have to be completed and unit tested before the integration process starts.