



POLITECNICO MILANO 1863

Politecnico di Milano
A.A. 2015/2016
Software Engineering 2

Assignment 4: Integration Test Plan

Version 1.0

Alessandro Baldassari (mat. 841561)
Alberto Bendin (mat. 841734)
Francesco Giarola (mat. 840554)

January 15, 2016

Contents

| | Page |
|---|----------|
| 1 Introduction | 1 |
| 1.1 Revision History | 1 |
| 1.2 Purpose and Scope | 1 |
| 1.3 List of Definitions and Abbreviations | 1 |
| 1.4 List of Reference Documents | 2 |
| 2 Integration Strategy | 2 |
| 2.1 Entry Criteria | 2 |
| 2.2 Elements to be Integrated | 3 |
| 2.3 Integration Testing Strategy | 4 |
| 2.4 Sequence of Component/Function Integration | 4 |
| 2.4.1 Sequence of integration for Account Manager sub-system | 4 |
| 2.4.2 Sequence of integration for Zone Manager sub-system | 4 |
| 2.4.3 Sequence of integration for Ride Manager sub-system | 5 |
| 2.4.4 Sequence of integration for Request Manager sub-system | 5 |
| 2.4.5 Sequence of integration for Client | 5 |
| 2.4.6 Sequence of integration for Request Manager | 6 |
| 2.4.7 Sequence of integration for Ride Manager | 6 |
| 2.4.8 Sequence of integration for Zone Manager | 6 |
| 3 Individual Steps and Test Description | 7 |
| 3.1 Test Case Specifications | 7 |
| 3.1.1 Integration tests for Account Manager | 7 |
| 3.1.2 Integration tests for Zone Manager | 8 |
| 3.1.3 Integration tests for Ride Manager | 9 |
| 3.1.4 Integration tests for Request Manager | 10 |
| 3.1.5 Integration tests for Client and Account Manager | 11 |
| 3.1.6 Integration tests for Client and Payment Manager | 11 |
| 3.1.7 Integration tests for Client and Request Manager | 12 |
| 3.1.8 Integration tests for Client and Ride Manager | 13 |
| 3.1.9 Integration tests for Client and Zone Manager | 14 |
| 3.1.10 Integration tests for Request Manager and Ride Manager | 15 |
| 3.1.11 Integration tests for Request Manager and Notification Manager | 15 |
| 3.1.12 Integration tests for Ride Manager and Zone Manager | 16 |
| 3.1.13 Integration tests for Ride Manager and Payment Manager | 16 |
| 3.1.14 Integration tests for Ride Manager and Notification Manager | 17 |
| 3.1.15 Integration tests for Zone Manager and Taxi Manager | 17 |
| 3.1.16 Integration tests for Zone Manager and Notification Manager | 18 |
| 3.2 Test procedures | 18 |
| 3.2.1 Test procedure for Account Manager sub-system | 18 |
| 3.2.2 Test procedure for Request Manager sub-system | 19 |
| 3.2.3 Test procedure for Ride Manager sub-system | 19 |
| 3.2.4 Test procedure for Zone Manager sub-system | 20 |

| | | | |
|----------|---|--|-----------|
| | 3.2.5 | Test procedure for components involved in User login | 21 |
| | 3.2.6 | Test procedure for components involved in Passenger request | 22 |
| | 3.2.7 | Test procedure for components involved in Request cancellation . | 23 |
| | 3.2.8 | Test procedure for components involved in Driver's job assignation | 24 |
| 4 | Tools and Test Equipment Required | | 25 |
| 5 | Program Stubs and Test Data Required | | 26 |
| | 5.1 | Stubs | 26 |
| | 5.2 | Drivers | 26 |
| | 5.3 | Data required | 26 |
| 6 | References | | 27 |
| 7 | Appendix | | 27 |
| | 7.1 | Software and tools used | 27 |
| | 7.2 | Hours of work | 27 |

1 Introduction

1.1 Revision History

This is the first version of the document. There are no previous versions.

| Revision | Last Edited | Changes |
|----------|-------------|--------------------|
| 1.0 | 21/01/2016 | Document redaction |

1.2 Purpose and Scope

The Test Plan Document (ITPD) describes the plan to accomplish the integration test. This document is supposed to be written before the integration test really happens and takes the architectural description of the software system as a starting point, for this reason it is often redacted in parallel with the Design Document. It explain to the development team what to test, in which sequence, which tools are needed for testing (if any), which stubs/drivers/oracles need to be developed.

The purpose of integration testing is to verify functional, performance, and reliability requirements of individual software modules of the product when they are combined and tested as a group; i.e., units (or groups of units) are exercised through their interfaces. The aim is to test the modules interactions incrementally, with success and error cases being simulated via appropriate parameter and data inputs. Simulated usage of shared data areas and inter-process communication is tested and individual subsystems are exercised through their input interface. Test cases are constructed to test whether all the components interact correctly, for example across procedure calls or process activations.

This is done after testing individual modules, i.e., unit testing; the overall idea is a "building block" approach, in which verified assemblages are added to a verified base which is then used to support the integration testing of further assemblages up to the complete final system (the testing on the complete system is not part of this integration testing phase).

1.3 List of Definitions and Abbreviations

The following acronyms are used in this document:

- RASD: Requirements Analysis and Specification Document
- DD: Design Document
- DB: DataBase. It is a test Database filled with test data. See section 5. [link](#)

The following definitions are used in this document:

- Driver: are considered dummy modules which are always distinguished as "calling programs that are handled in bottom up integration testing; they are only used when main programs are under construction.
- Stub: in computer science, test stubs are programs that simulate the behaviors of software components (or modules) that a module undergoing tests depends on. Test stubs are mainly used in incremental testing's top-down approach.
- Oracle: in computing, software testers and software engineers can use an oracle as a mechanism for determining whether a test has passed or failed. The use of oracles involves comparing the output(s) of the system under test, for a given test-case input, to the output(s) that the oracle determines that product should have.

1.4 List of Reference Documents

- Specification document: myTaxiService project
- Requirements Analysis and Specification Document (RASD) for myTaxiService
- Design Document (DD) for myTaxiService

2 Integration Strategy

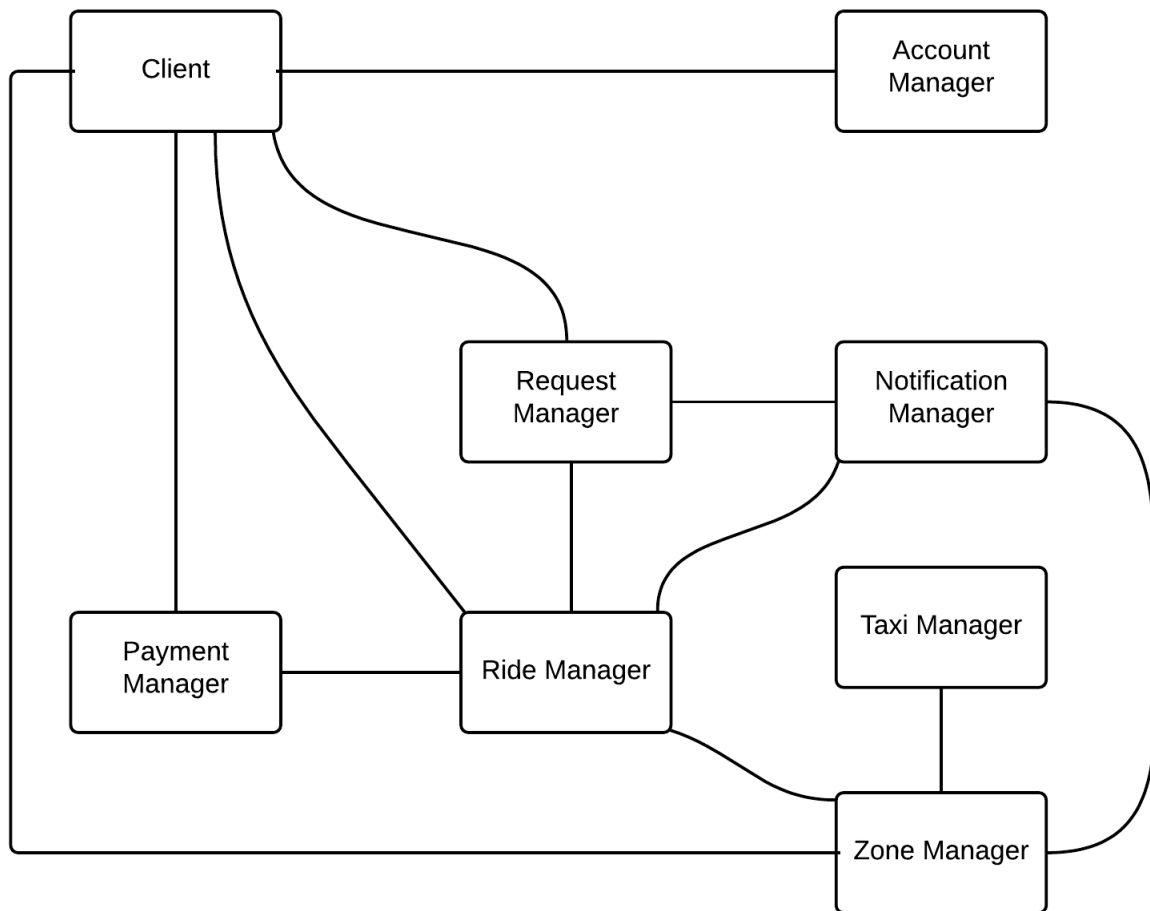
2.1 Entry Criteria

It is supposed that the unit testing phase has already been completed successfully. In particular the following components are considered already tested and thus trusted as working:

- Payment Manager
- Notification Manager
- Taxi Manager
- DB
- Google Maps APIs

2.2 Elements to be Integrated

The scheme below show the main high level components of the system.



From test I1 up to test I7 the integration is related to sub-systems.

Starting from I8 tests will be focused on integration of higher level components.

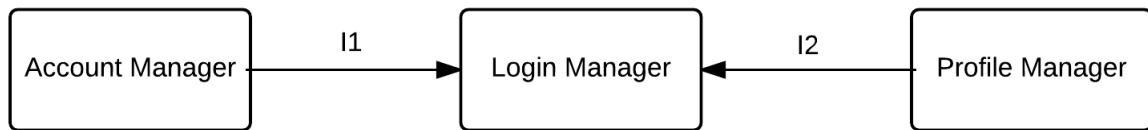
| ID | Integration Test | Paragraphs | | |
|-----|---------------------------------------|------------|--------|-------|
| I1 | Login Manager →Account Factory | 2.4.1 | 3.1.1 | 3.2.1 |
| I2 | Profile Manager →Login Manager | 2.4.1 | 3.1.1 | 3.2.1 |
| I3 | Zone Engine →Queue Manager | 2.4.2 | 3.1.2 | 3.2.4 |
| I4 | Ride Handler →Ride Worker | 2.4.3 | 3.1.3 | 3.2.3 |
| I5 | Ride Worker →Ride Engine | 2.4.3 | 3.1.3 | 3.2.3 |
| I6 | Request Handler →Request Worker | 2.4.4 | 3.1.4 | 3.2.2 |
| I7 | Request Worker →Request Engine | 2.4.4 | 3.1.4 | 3.2.2 |
| I8 | Client →Account Manager | 2.4.5 | 3.1.5 | |
| I9 | Client →Payment Manager | 2.4.5 | 3.1.6 | |
| I10 | Client →Request Manager | 2.4.5 | 3.1.7 | |
| I11 | Client →Ride Manager | 2.4.5 | 3.1.8 | |
| I12 | Client →Zone Manager | 2.4.5 | 3.1.9 | |
| I13 | Request Manager →Ride Manager | 2.4.6 | 3.1.10 | |
| I14 | Request Manager →Notification Manager | 2.4.6 | 3.1.11 | |
| I15 | Ride Manager →Zone Manager | 2.4.7 | 3.1.12 | |
| I16 | Ride Manager →Payment Manager | 2.4.7 | 3.1.13 | |
| I17 | Ride Manager →Notification Manager | 2.4.7 | 3.1.14 | |
| I18 | Zone Manager →Taxi Manager | 2.4.8 | 3.1.15 | |
| I19 | Zone Manager →Notification Manager | 2.4.8 | 3.1.16 | |

2.3 Integration Testing Strategy

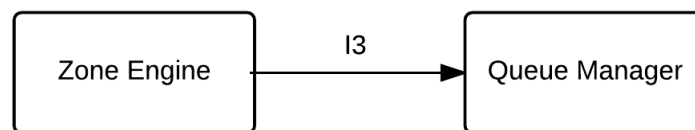
An hybrid approach will be used to accomplish the analysis. At the beginning the integration inside the sub-systems of the high level components will be taken into account and consequently the integration between the high level components themselves.

2.4 Sequence of Component/Function Integration

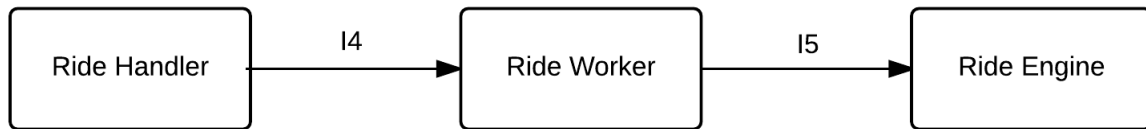
2.4.1 Sequence of integration for Account Manager sub-system



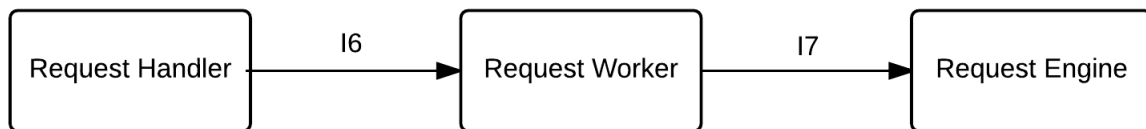
2.4.2 Sequence of integration for Zone Manager sub-system



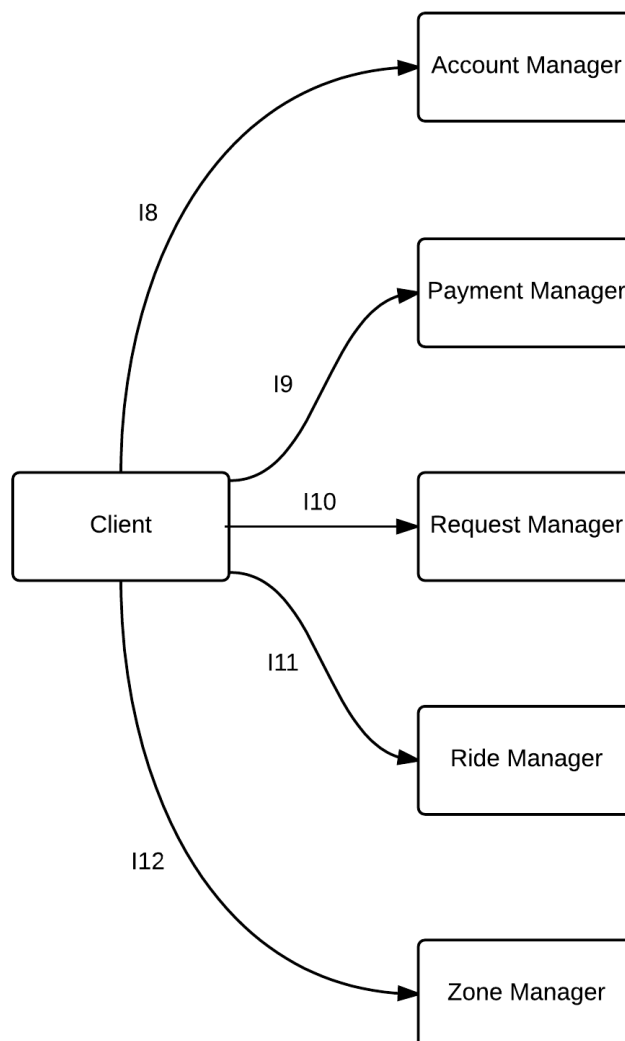
2.4.3 Sequence of integration for Ride Manager sub-system



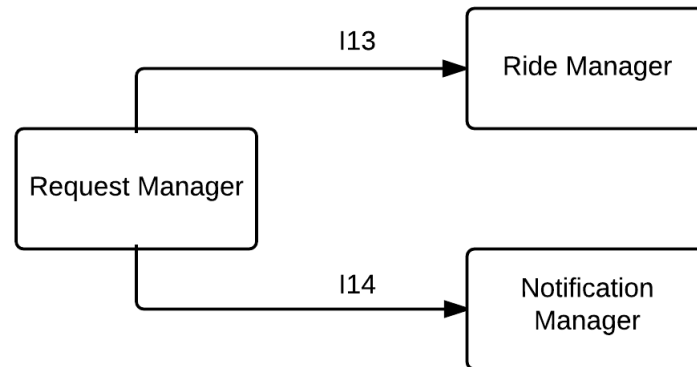
2.4.4 Sequence of integration for Request Manager sub-system



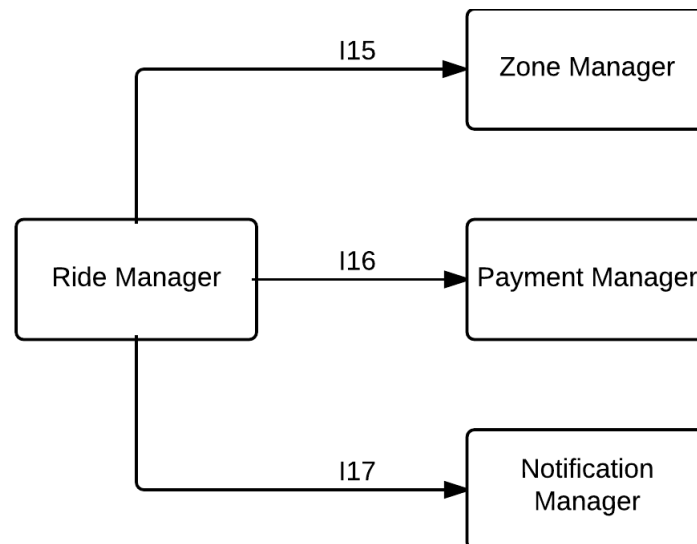
2.4.5 Sequence of integration for Client



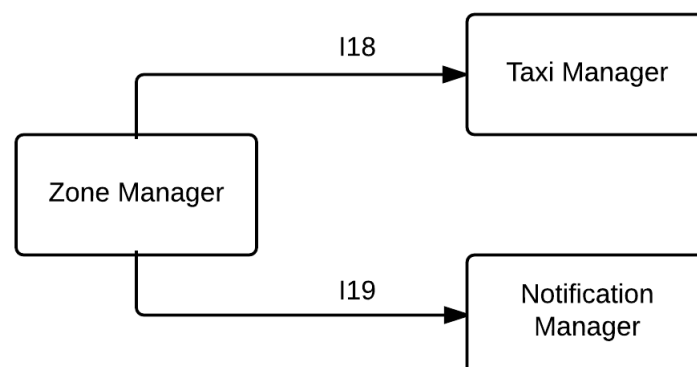
2.4.6 Sequence of integration for Request Manager



2.4.7 Sequence of integration for Ride Manager



2.4.8 Sequence of integration for Zone Manager



3 Individual Steps and Test Description

3.1 Test Case Specifications

3.1.1 Integration tests for Account Manager

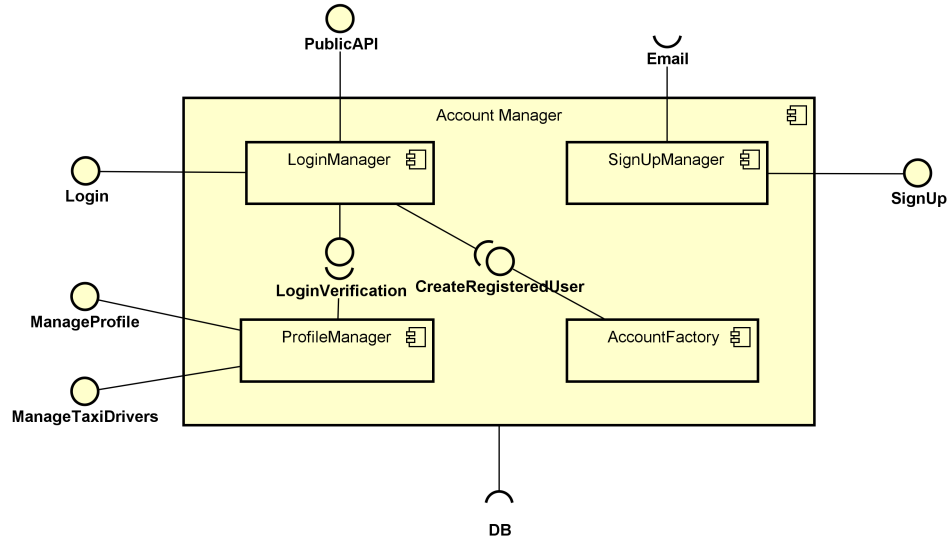


Figure taken from paragraph 2.3.1 of the DD

| | |
|-----------------------------|--|
| Test case identifier | I1 |
| Test item(s) | Login Manager → Account Factory |
| Input specification | Create typical Login Manager input |
| Output specification | Check if the correct methods are called in the Account Factory |
| Environmental needs | Client driver and the DB |

| | |
|-----------------------------|--|
| Test case identifier | I2 |
| Test item(s) | Profile Manager → Login Manager |
| Input specification | Create typical Profile Manager input |
| Output specification | Check if the correct functions are called in the Login Manager |
| Environmental needs | Client driver and the DB |

3.1.2 Integration tests for Zone Manager

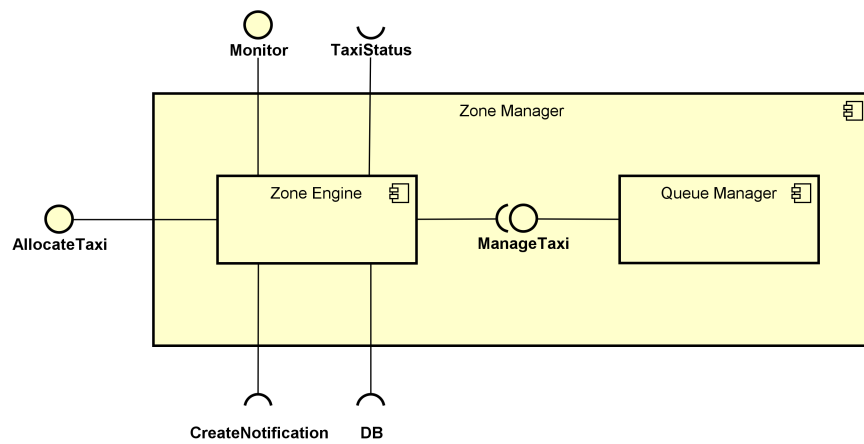


Figure taken from paragraph 2.3.6 of the DD

| | |
|-----------------------------|--|
| Test case identifier | I3 |
| Test item(s) | Zone Engine → Queue Manager |
| Input specification | Create typical Zone Engine input |
| Output specification | Check if the correct functions are called in the Queue Manager |
| Environmental needs | Ride Manager driver |

3.1.3 Integration tests for Ride Manager

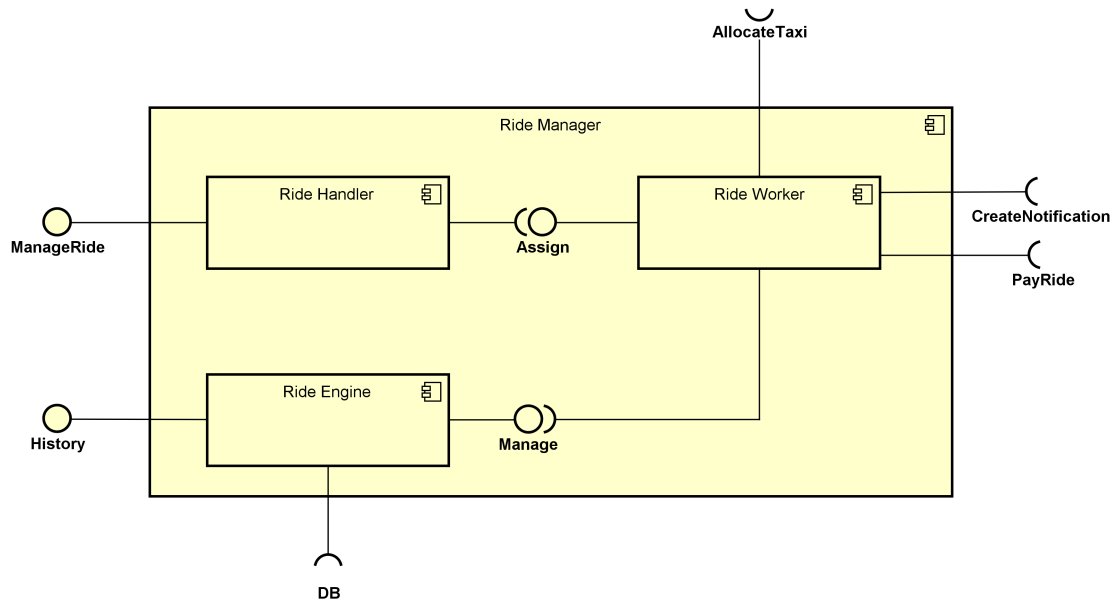


Figure taken from paragraph 2.3.3 of the DD

| | |
|-----------------------------|--|
| Test case identifier | I4 |
| Test item(s) | Ride Handler → Ride Worker |
| Input specification | Create typical Ride Handler input |
| Output specification | Check if the correct functions are called in the Ride Worker |
| Environmental needs | Request Manager driver to simulate a typical communication input (creation or modification of ride), the stubs for Payment Manager, Zone Manager and Notification Manager to satisfy the components inter-dependencies |

| | |
|-----------------------------|--|
| Test case identifier | I5 |
| Test item(s) | Ride Worker → Ride Engine |
| Input specification | Create typical Ride Worker input |
| Output specification | Check if the correct functions are called in the Ride Engine |
| Environmental needs | I4 succeeded, the DB |

3.1.4 Integration tests for Request Manager

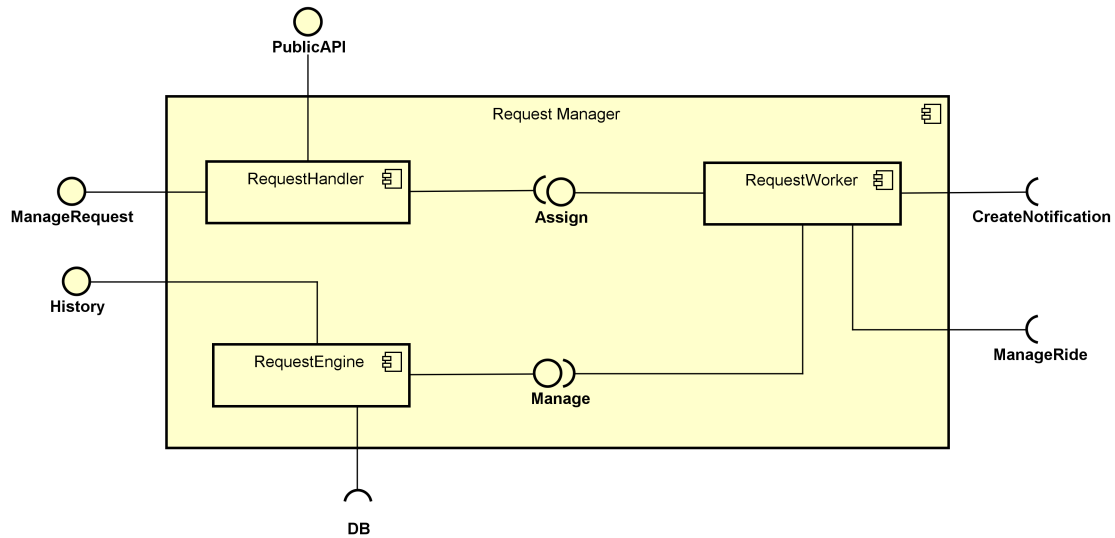
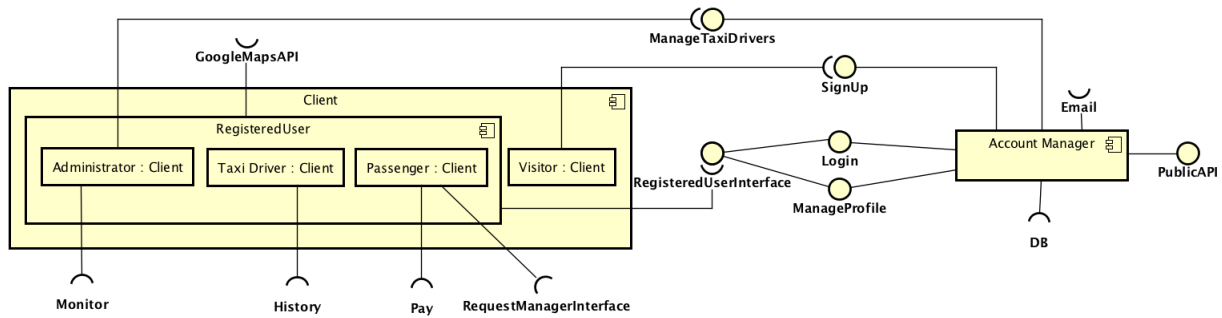


Figure taken from paragraph 2.3.2 of the DD

| | |
|----------------------|---|
| Test case identifier | I6 |
| Test item(s) | Request Handler →Request Worker |
| Input specification | Create typical Request Handler input |
| Output specification | Check if the correct functions are called in the Request Worker |
| Environmental needs | Client (Passenger) driver to simulate a typical communication input (creation or modification of request/reservation), the stubs for Ride Manager and Notification Manager to satisfy the components inter-dependencies |

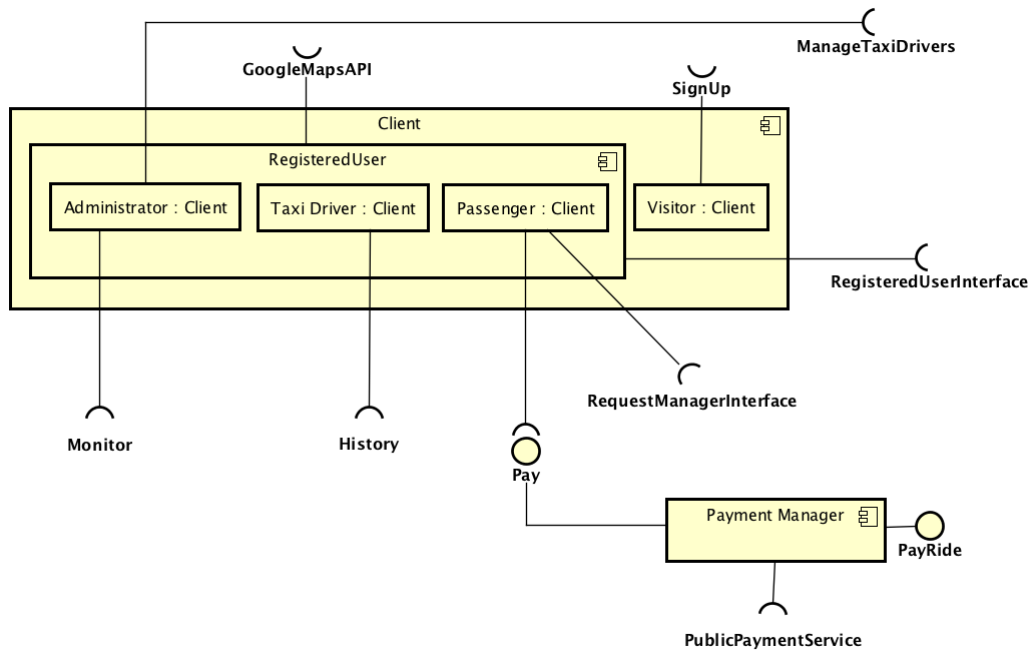
| | |
|----------------------|---|
| Test case identifier | I7 |
| Test item(s) | Request Worker →Request Engine |
| Input specification | Create typical Request Worker input |
| Output specification | Check if the correct functions are called in the Request Engine |
| Environmental needs | I6 succeeded, the DB |

3.1.5 Integration tests for Client and Account Manager



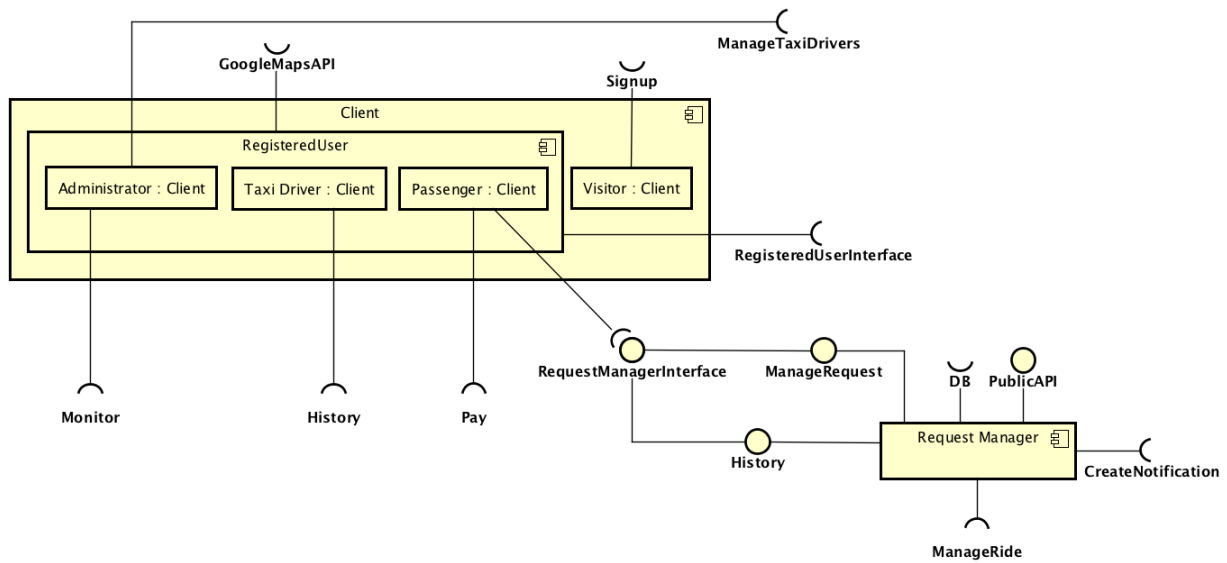
| | |
|----------------------|--|
| Test case identifier | I8 |
| Test item(s) | Client → Account Manager |
| Input specification | Create typical Client input |
| Output specification | Check if the correct functions are called in the Account Manager |
| Environmental needs | I1, I2 succeeded, the DB, the stub for Notification Manager to satisfy the components inter-dependencies |

3.1.6 Integration tests for Client and Payment Manager



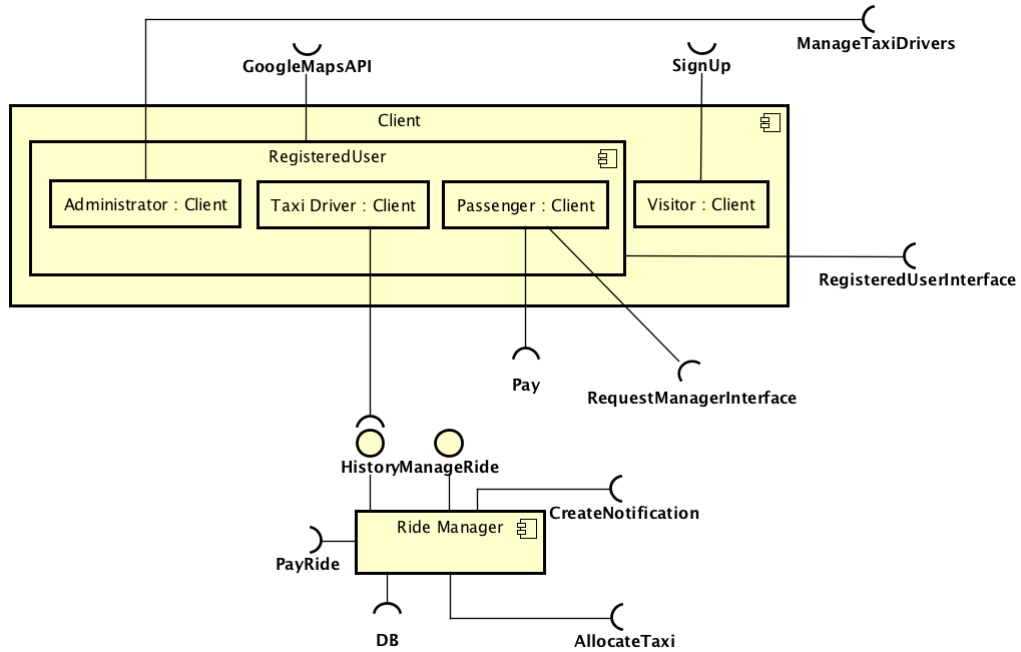
| | |
|----------------------|--|
| Test case identifier | I9 |
| Test item(s) | Client → Payment Manager |
| Input specification | Create typical Client input |
| Output specification | Check if the correct functions are called in the Payment Manager |
| Environmental needs | The stub for Public Payment Service to satisfy the components inter-dependencies |

3.1.7 Integration tests for Client and Request Manager



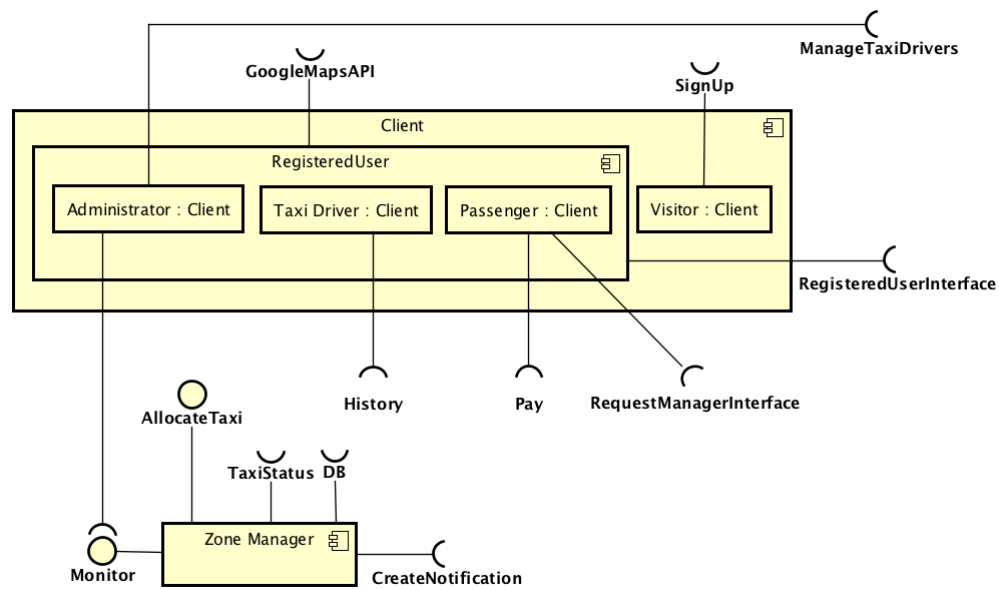
| | |
|-----------------------------|--|
| Test case identifier | I10 |
| Test item(s) | Client →Request Manager |
| Input specification | Create typical Client input |
| Output specification | Check if the correct functions are called in the Request Manager |
| Environmental needs | I6, I7 succeeded, the DB and the stubs for Ride Manager, Notification Manager to satisfy the components inter-dependencies |

3.1.8 Integration tests for Client and Ride Manager



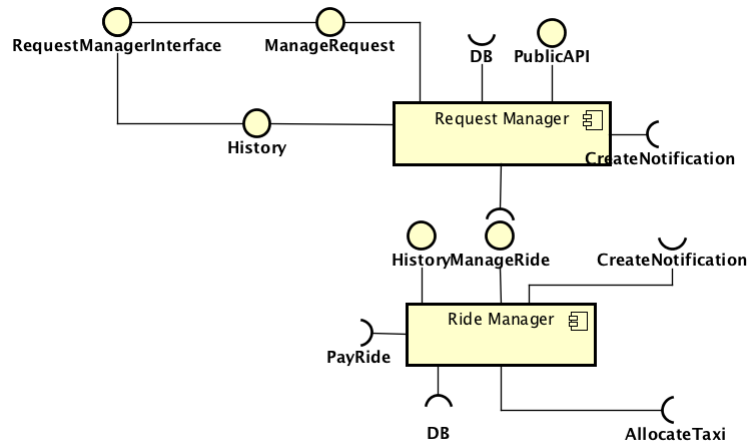
| | |
|-----------------------------|---|
| Test case identifier | I11 |
| Test item(s) | Client → Ride Manager |
| Input specification | Create typical Client input |
| Output specification | Check if the correct functions are called in the Ride Manager |
| Environmental needs | I4, I5 succeeded, the DB and the stubs for Payment Manager, Zone Manager, Notification Manager to satisfy the components inter-dependencies |

3.1.9 Integration tests for Client and Zone Manager



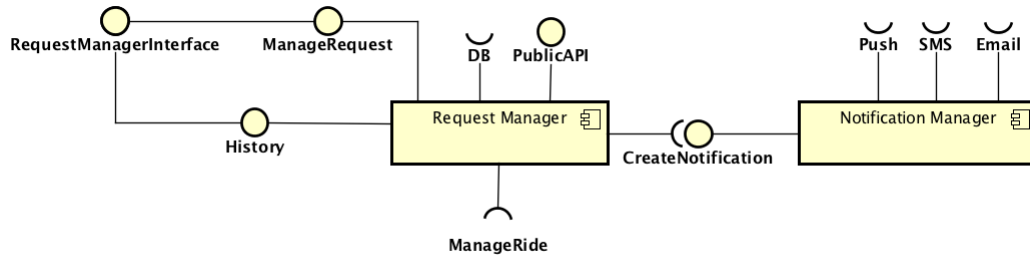
| | |
|-----------------------------|--|
| Test case identifier | I12 |
| Test item(s) | Client →Zone Manager |
| Input specification | Create typical Client input |
| Output specification | Check if the correct functions are called in the Zone Manager |
| Environmental needs | I3 succeeded, the DB and the stubs for Taxi Manager, Notification Manager to satisfy the components inter-dependencies |

3.1.10 Integration tests for Request Manager and Ride Manager



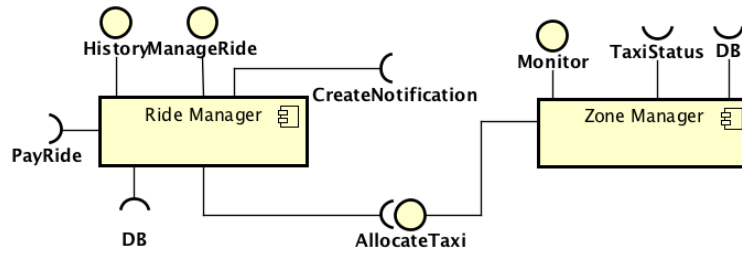
| | |
|----------------------|--|
| Test case identifier | I13 |
| Test item(s) | Request Manager → Ride Manager |
| Input specification | Create typical Request Manager input |
| Output specification | Check if the correct functions are called in the Ride Manager |
| Environmental needs | I4, I5, I6, I7 succeeded, the DB, the stubs for Notification Manager, Zone Manager, Payment Manager to satisfy the components inter-dependencies |

3.1.11 Integration tests for Request Manager and Notification Manager



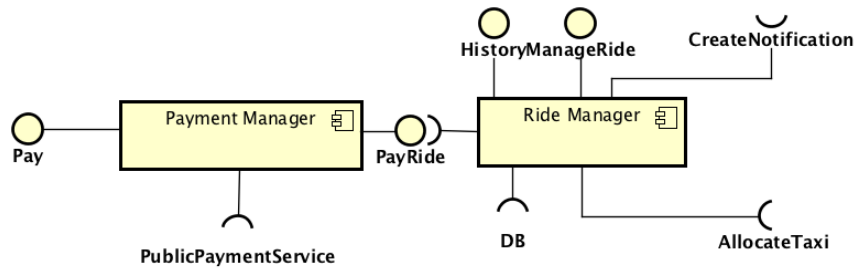
| | |
|----------------------|--|
| Test case identifier | I14 |
| Test item(s) | Request Manager → Notification Manager |
| Input specification | Create typical Request Manager input |
| Output specification | Check if the correct functions are called in the Notification Manager |
| Environmental needs | I6, I7 succeeded, the stubs for SMS, Email, Push notification to satisfy the components inter-dependencies |

3.1.12 Integration tests for Ride Manager and Zone Manager



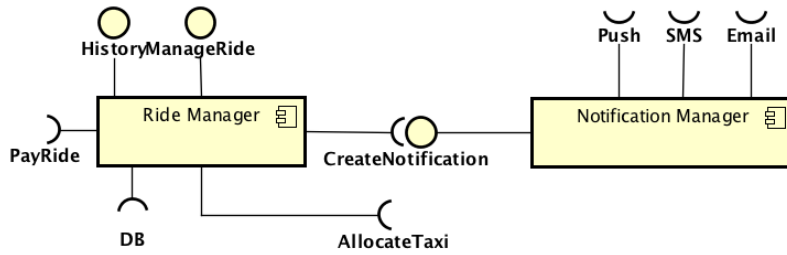
| | |
|----------------------|--|
| Test case identifier | I15 |
| Test item(s) | Ride Manager →Zone Manager |
| Input specification | Create typical Ride Manager input |
| Output specification | Check if the correct functions are called in the Zone Manager |
| Environmental needs | I3, I4, I5 succeeded, the DB, the stub Taxi Manager to satisfy the components inter-dependencies |

3.1.13 Integration tests for Ride Manager and Payment Manager



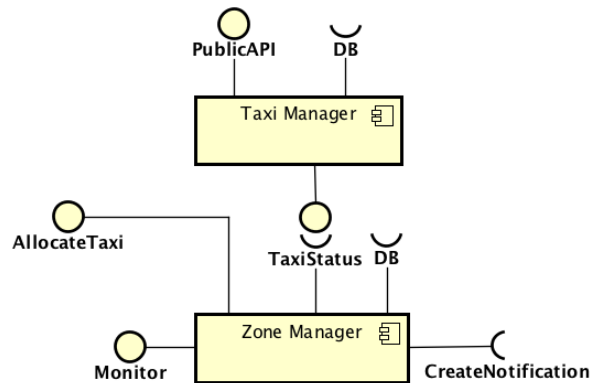
| | |
|----------------------|--|
| Test case identifier | I16 |
| Test item(s) | Ride Manager →Payment Manager |
| Input specification | Create typical Ride Manager input |
| Output specification | Check if the correct functions are called in the Payment Manager |
| Environmental needs | I4, I5 succeeded, the stub for Public Payment Service to satisfy the components inter-dependencies |

3.1.14 Integration tests for Ride Manager and Notification Manager



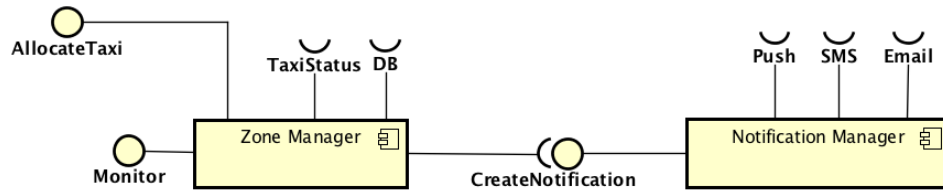
| | |
|----------------------|--|
| Test case identifier | I17 |
| Test item(s) | Ride Manager → Notification Manager |
| Input specification | Create typical Ride Manager input |
| Output specification | Check if the correct functions are called in the Notification Manager |
| Environmental needs | I4, I5 succeeded, the stubs for SMS, Email, Push notification to satisfy the components inter-dependencies |

3.1.15 Integration tests for Zone Manager and Taxi Manager



| | |
|----------------------|---|
| Test case identifier | I18 |
| Test item(s) | Zone Manager → Taxi Manager |
| Input specification | Create typical Zone Manager input |
| Output specification | Check if the correct functions are called in the Taxi Manager |
| Environmental needs | I3 succeeded, the DB |

3.1.16 Integration tests for Zone Manager and Notification Manager



| | |
|-----------------------------|--|
| Test case identifier | I19 |
| Test item(s) | Zone Manager → Notification Manager |
| Input specification | Create typical Zone Manager input |
| Output specification | Check if the correct functions are called in the Notification Manager |
| Environmental needs | I3 succeeded, the stubs for SMS, Email, Push notification to satisfy the components inter-dependencies |

3.2 Test procedures

3.2.1 Test procedure for Account Manager sub-system

| | |
|----------------------------------|---|
| Test procedure identifier | TP1 |
| Purpose | <p>This test verifies that the Account Manager component:</p> <ul style="list-style-type: none"> • Can handle the Visitor input • Can handle the Registered User input • Can output the requested information to a Visitor • Can output the requested information to a Registered User • Can retrieve the correct information about a user |
| Procedure steps | Execute I2 after I1 |

3.2.2 Test procedure for Request Manager sub-system

| | |
|----------------------------------|---|
| Test procedure identifier | TP2 |
| Purpose | <p>This test verifies that the Request Manager component:</p> <ul style="list-style-type: none">• Can handle the Passenger input (creation/modification of requests/reservations and user's history)• Can handle the Passenger input from an external service through the APIs• Can output the requested information to a Passenger• Can start a thread for every user's request• Can modify and delete requests, thus it can retrieve information about the requests from the DB• Can retrieve the correct information about a user |
| Procedure steps | Execute I7 after I6 |

3.2.3 Test procedure for Ride Manager sub-system

| | |
|----------------------------------|--|
| Test procedure identifier | TP3 |
| Purpose | <p>This test verifies that the Ride Manager component:</p> <ul style="list-style-type: none">• Can handle the creation, modification and cancellation of rides• Can start a thread for every user's ride• Can handle the Taxi driver's input• Can output the requested information to a Taxi driver• Can retrieve information about the rides from the DB• Can retrieve the correct information about a user• Can launch a payment process |
| Procedure steps | Execute I5 after I4 |

3.2.4 Test procedure for Zone Manager sub-system

| | |
|----------------------------------|---|
| Test procedure identifier | TP4 |
| Purpose | <p>This test verifies that the Zone Manager component:</p> <ul style="list-style-type: none">• Can handle the Administrator input• Can output the requested information to the Administrator• Can manage the queues of taxis in the zones• Can forward a request to the correct queue• Can retrieve information about the queues from the DB• Can retrieve the correct information about a specific Taxi driver• Can send job proposals to taxi drivers |
| Procedure steps | Execute I3 |

3.2.5 Test procedure for components involved in User login

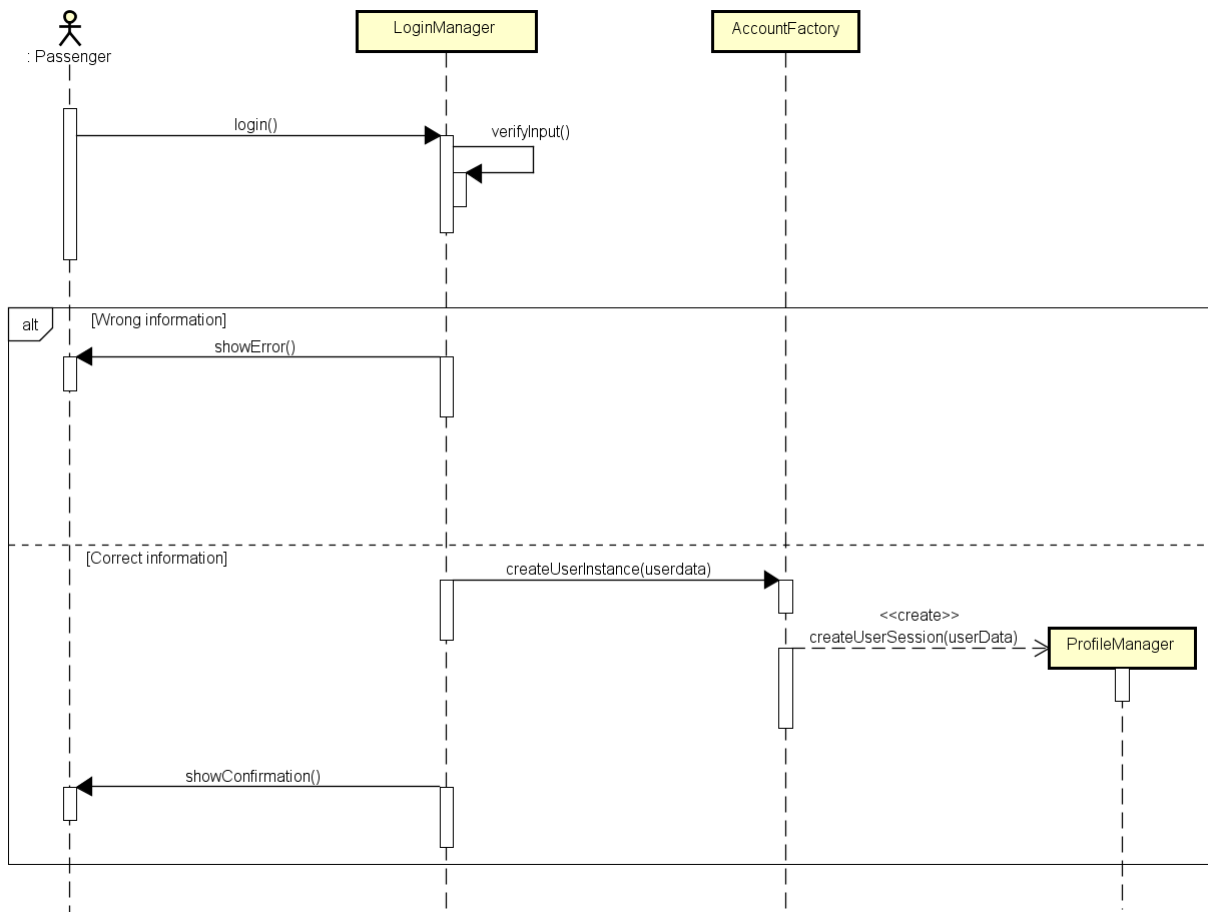


Figure taken from paragraph 2.5.2.1 of the DD

| | |
|----------------------------------|--|
| Test procedure identifier | TP5 |
| Purpose | This test verifies the login procedure in order to check the correct integration of all the involved components. |
| Procedure steps | Execute in order I1 and I2 |

3.2.6 Test procedure for components involved in Passenger request

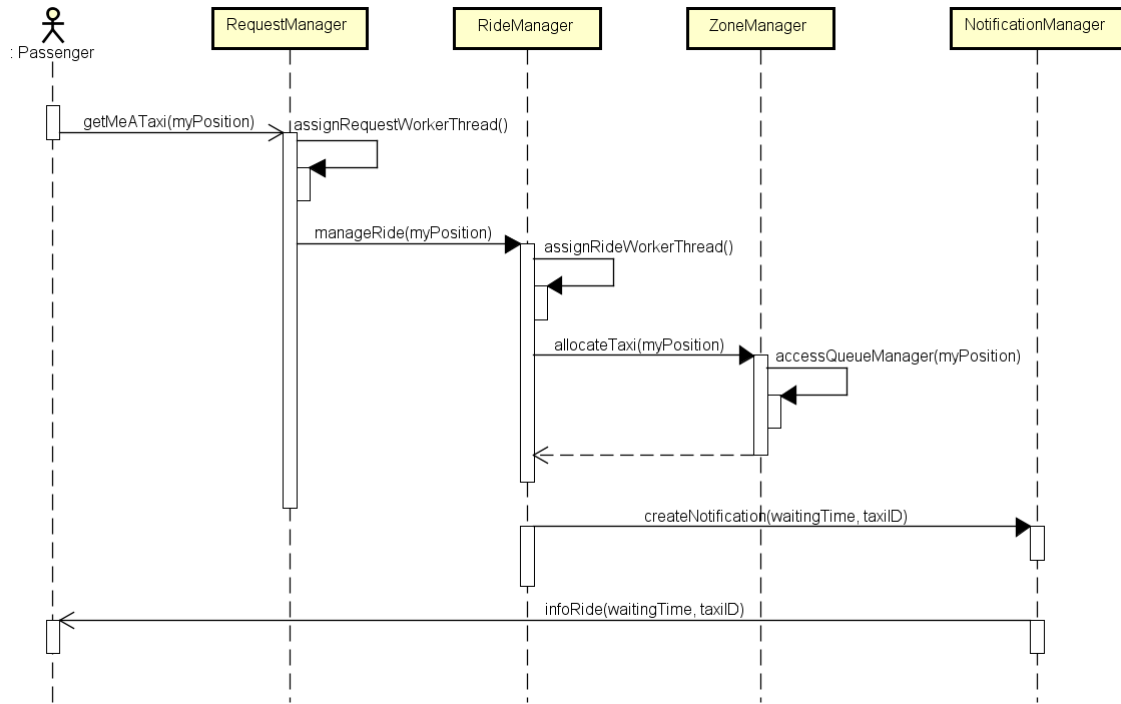


Figure taken from paragraph 2.5.2.2 of the DD

| | |
|----------------------------------|---|
| Test procedure identifier | TP6 |
| Purpose | This test verifies the procedure activated when a passenger asks for a taxi in order to check the correct integration of all the involved components. |
| Procedure steps | Execute in order I10, I13, I15 and I17 |

3.2.7 Test procedure for components involved in Request cancellation

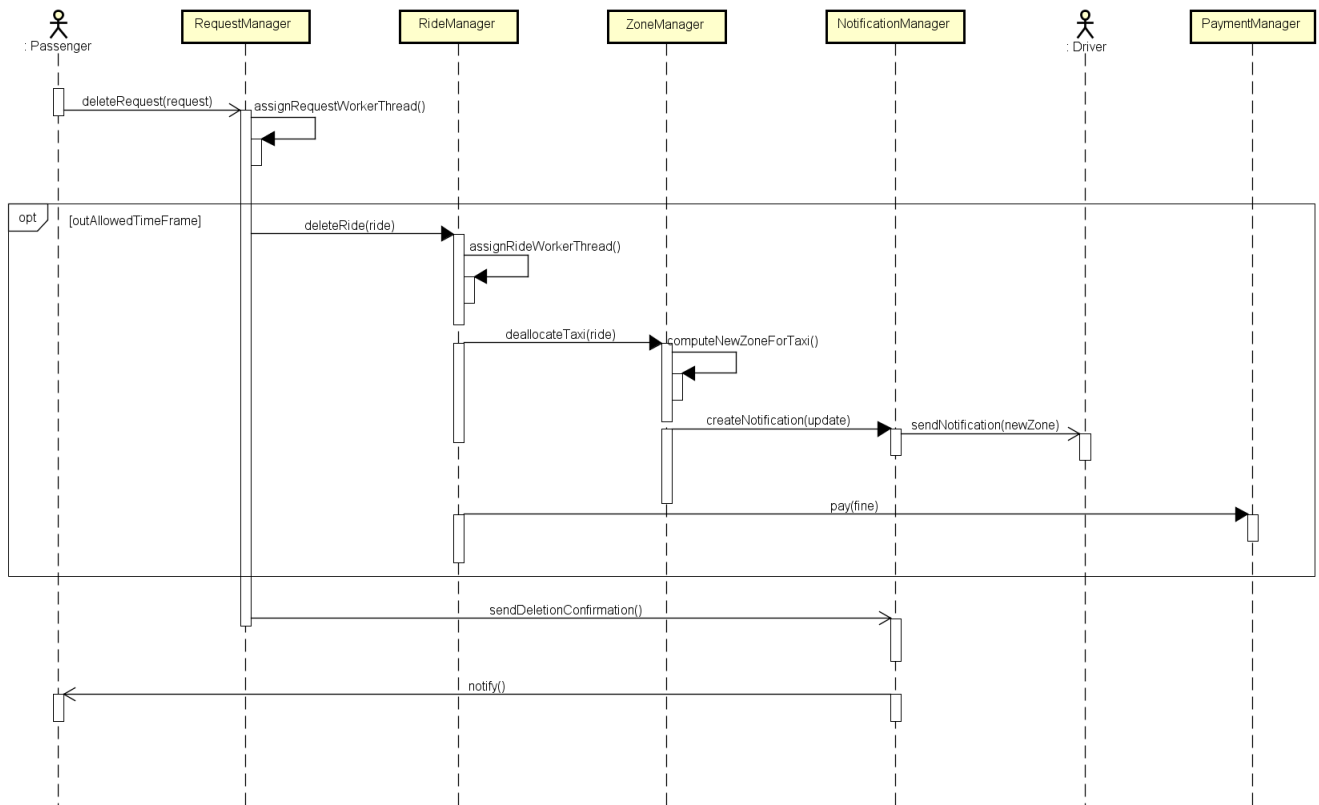


Figure taken from paragraph 2.5.2.3 of the DD

| | |
|----------------------------------|---|
| Test procedure identifier | TP7 |
| Purpose | This test verifies the procedure activated when a passenger deletes a request or reservation for a taxi (when the [outAllowedTimeFrame] condition is not satisfied) in order to check the correct integration of all the involved components. |
| Procedure steps | Execute in order I10 and I14 |

| | |
|----------------------------------|---|
| Test procedure identifier | TP8 |
| Purpose | This test verifies the procedure activated when a passenger deletes a request or reservation for a taxi (when the [outAllowedTimeFrame] condition is satisfied) in order to check the correct integration of all the involved components. |
| Procedure steps | Execute in order I10, I13, I15, I19, I16 and I14 |

3.2.8 Test procedure for components involved in Driver's job assignment

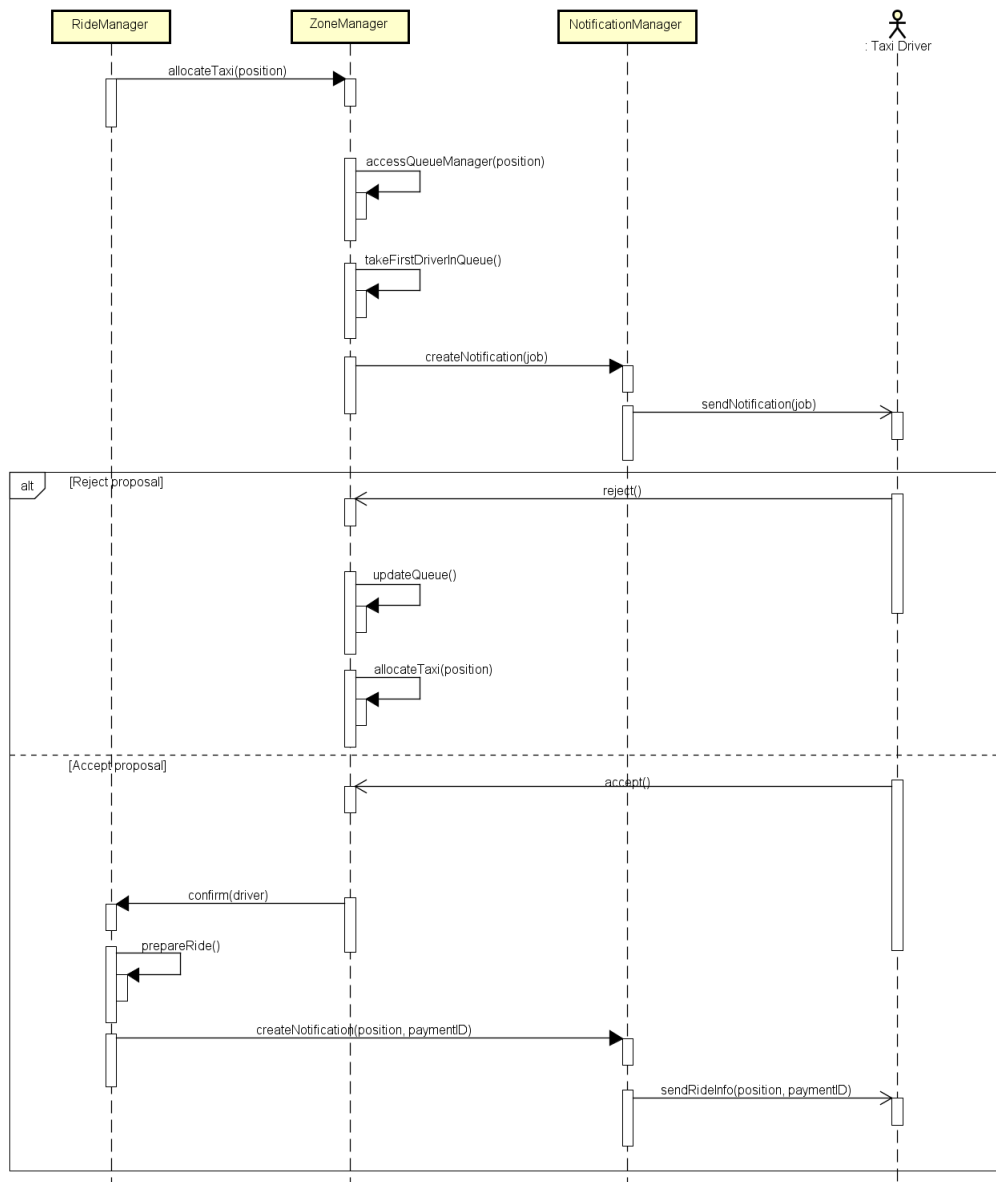


Figure taken from paragraph 2.5.2.4 of the DD

| | |
|----------------------------------|---|
| Test procedure identifier | TP9 |
| Purpose | This test verifies the procedure activated when the system assigns a job to a driver (when the [Reject proposal] branch is taken) in order to check the correct integration of all the involved components. |
| Procedure steps | Execute in order I15 and I19 |

| | |
|----------------------------------|---|
| Test procedure identifier | TP10 |
| Purpose | This test verifies the procedure activated when the system assigns a job to a driver (when the [Accept proposal] branch is taken) in order to check the correct integration of all the involved components. |
| Procedure steps | Execute in order I15, I19 and I17 |

4 Tools and Test Equipment Required

To perform in a reliable way the integration testing, the following tools and testing environments are needed. In order to accomplish the integration testing at sub-component level (e.g. integration of the components in the Account Manager) we are going to use the integration framework of Mockito's library. We made this choice because this tool offers the possibility to implement the stubs and the drivers needed for the testing. We are going to use Mockito also in the integration testing of mobile clients.

To perform the integration testing at component level (e.g. integration of Zone Manager with Ride Manager) we are going to use the Arquillian framework. This is due to the fact that we have several runtime units that are going to run in different software containers inside the same virtual machine and this framework provides the tools to test software containers integration. We take in consideration manual testing to test the integration between the web client and the other components of the system.

The testing environment consists in a virtual machine for the application logic and another one for the DBMS running both Ubuntu Server 14.04 LTS. Concerning the client side we need a mobile terminal running Android 6.0.1, another mobile terminal running iOS 9.2 and a desktop PC running Google Chrome.

5 Program Stubs and Test Data Required

5.1 Stubs

- Payment Manager
- Zone Manager
- Notification Manager
- Ride Manager
- Public Payment Service
- Taxi Manager
- SMS service
- Mail service
- Push service

5.2 Drivers

- Client (Administrator, Taxi Driver, Passenger)
- Ride Manager
- Request Manager

5.3 Data required

To perform a correct integration testing it is necessary to implement a dummy DB populated with the following data:

- Some user data for each user type (Administrator, Taxi Driver, Passenger)
- A set of requests for shared and non-shared trips
- A set of rides
- A set of city zones
- A set of taxis to populate each zones taxi queue

All this data elements must match together to perform correctly the integration testing. In order for a request and its associated ride to be correct, at least a taxi driver and one or more passengers should appear. The taxi driver should match with his taxi and the taxi should be in the starting zone of the ride.

6 References

Material from Wikipedia

- Integration testing: https://en.wikipedia.org/wiki/Integration_testing
- Oracles: [https://en.wikipedia.org/wiki/Oracle_\(software_testing\)](https://en.wikipedia.org/wiki/Oracle_(software_testing))

7 Appendix

7.1 Software and tools used

- TeXstudio 2.10.6 (<http://www.texstudio.org/>) to redact and format this document.
- Astah Professional 7.0 (<http://astah.net/editions/professional>)

7.2 Hours of work

The time spent to redact this document:

- Baldassari Alessandro: 20 hours.
- Bendin Alberto: 20 hours.
- Giarola Francesco: 20 hours.