INFORMATION TECHNOLOGIES FOR SHIFT TO RAIL

D3.6 – Booking and Ticketing Final Integration Report

Due date of deliverable: 30/11/2017

Actual submission date: 28/06/2018

Leader/Responsible of this Deliverable: THALES

Reviewed: Y

Document status		
Revision	Date	Description
0.1	06/12/2017	Circulated for Comments
0.2	26/03/2017	Sent for Review
0.3	30/04/2018	Other tests performed and contributions received
0.4	08/05/2018	Updates for Urban
0.5	25/05/2018	Removed prefix of partners in naming of tests
1	30/05/2018	Slight updates for Air ‘pushToCloud’ & Rail configuration
2	20/05/2018	Final version for TMC approval
3	28/06/2018	Final version after TMC approval and Quality check

Project funded from the European Union’s Horizon 2020 research and innovation programme		
Dissemination Level		
PU	Public	X
CO	Confidential, restricted under conditions set out in Model Grant Agreement	
CI	Classified, information as referred to in Commission Decision 2001/844/EC	

Start date of project: 01/05/2015 Duration: 36 months

EXECUTIVE SUMMARY

This document details the Test Categories and Test Cases identified by the partners for the Final Release of WP3. For each of the test cases identified, a description is included detailing the objectives, expected results and how to perform the testing.

The objectives of this test campaign in particular are to perform unit testing of each of the modules of WP3, as well as to test the interfaces with other WP3 modules and other WPs. The results obtained for each of the test identified and described here are also included in this document.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

This document details the Test Categories and Test Cases identified by the partners for the Final Release of WP3. For each of the test cases identified, a description is included detailing the objectives, expected results and how to perform the testing.

2. REFERENCED DOCUMENTS

2.1 APPLICABLE DOCUMENTS

IT2Rail	D7.2a – Development Readiness Review Pack
Indra	 INDRA - Rail TSP API v2.5

Table 1: Documents

2.1 NORMATIVE DOCUMENTS

Not applicable.

3. CAMPAIGN STRATEGY

This chapter describes the objective of the test campaigns covered by this document.

The following test campaigns are restricted to WP3 scope and serve for both unit testing and functional testing of WP3 elements of IT2Rail Final release.

WP3 Functional tests are the following:

- Orchestrate multiple Entitlement Issuing (with Air backend)
- Orchestrate Booking (simulating booking data and input)
- Air booking
- Air issuing of entitlement
- Rail Issuing of Entitlement
- Rail Issuing of Token
- Urban Issuing of entitlement
- Urban Issuing of Token

4. TEST MATERIAL DESCRIPTION

This chapter lists all the assets required to perform the test campaign. It is organised by configuration. Each test is associated with a particular configuration describing the tested system, its parameter and needed resources to conduct the tests.

4.1 CONFIGURATION ORCHESTRATORS

This configuration is the Amadeus orchestrator system including air backend. It handles TSP orchestration. Any orchestration internal to TSP is not covered by the provided system.

4.1.1 Infrastructure and Hardware

The system consists in web services running at Amadeus premises.

4.1.2 Setup & Configuration

A Non-Disclosure Agreement (NDA) should be signed.

Each client willing to invoke a web service should provide:

- IP address
- Port number
- Name phone and email of the network responsible

All the technical documentation required to connect the Amadeus system will be provided in the Amadeus web service portal at https://webservices.amadeus.com. Credentials will be provided on-demand.

Amadeus web services use SOAP protocol over HTTP/HTTPS.

4.1.3 Tested System

For ticket issuance:

Ticket_OrchestrateDeals version 4.1

For booking:

TTR_ManageTrip version 7.1

4.1.4 System Data Parameters

No parameter.

4.1.5 Simulators

No simulator in this configuration.

4.1.6 Personnel

No personnel is required for this configuration.

4.2 CONFIGURATION WP3-RAIL-01

This configuration is the Indra rail booking and issuing system.

4.2.1 Infrastructure and Hardware

The system consists in web services running at Indra premises.

Structure of the web service is shared in the referenced document “Rail-TSP_API_v2.5”.

4.2.2 Setup & Configuration

VPN client to be installed: FORTI client version 5.4.1.

Organisation contact should be provided to distribute VPN credentials.

Indra web services use SOAP protocol over HTTP/HTTPS.

The following action will need to be performed:

- Loading of topology
- Loading of fare rules
- Loading of dictionaries (operational parameter)
- Load customer
- Load business entities

4.2.3 Tested System

Rail Booking web services:

- ManageBookingI:InventoryLock
- ManageBookingI:ConfirmBooking
- ManageBookingI:IssueEntitlement
- ManageBookingI:IssueToken

Rail Manage offers web services:

- ManageOfferItemI:GetPrice

4.2.4 System Data Parameters

None at the moment.

4.2.5 Simulators

SOAPUI version 5.2.1

4.2.6 Personnel

No personnel is required for this configuration.

4.3 CONFIGURATION WP3-RAIL-02

This configuration is the SNCF rail booking and issuing/printing system.

4.3.1 Infrastructure and Hardware

The system consists in web services running at SNCF premises.

4.3.2 Setup & Configuration

Each client willing to invoke a web service should provide:

- IP address
- Login/password

All the technical documentation required to connect the SNCF rail booking and issuing/printing system is available online at
https://doc-vsct.vsct.fr/display/PUGP/PAO+User+Guide+Portal+Home

SNCF rail booking and issuing/printing system provides a RESTful API.
Credentials will be provided on-demand.

4.3.3 Tested System

Rail Booking web services:

- /sales/searchSolutions
- /sales/bookProposals

Rail Issuing/printing web services:

- /sales/createSalesContract
- /sales/printOrder

4.3.4 System Data Parameters

No parameter.

4.3.5 Simulators

No simulator in this configuration.

4.3.6 Personnel

No personnel is required for this configuration.

4.4 CONFIGURATION WP3-LDB01

This configuration is the OLTIS long distance buses booking and issuing system.

4.4.1 Infrastructure and Hardware

The system consists in API running at OLTIS premises.

4.4.2 Setup & Configuration

OLTIS API uses GET/POST methods over HTTP protocol.

4.4.3 Tested System

OLTIS long distance buses booking and issuing system are accessible via API through public internet.

4.4.4 System Data Parameters

No parameter.

4.4.5 Simulators

No simulator in this configuration.

4.4.6 Personnel

No personnel is required for this configuration.

4.5 CONFIGURATION WP3-URBAN01

This configuration is the Thales provided urban system.

4.5.1 Infrastructure and Hardware

Thales IT2Rail test platform is based on 2 VMs (virtual machines):

- "TUP": Transcity Back-Office
- "ITR": IT2Rail Gateway

The hardware that hosts "TUP" VM is a PC "HP ZBook" (model 17 G2) with these specifications:

HostName	CPU	Memory (Go)	Hard disk (Go)	OS
ITR (192.168.55.1)	8	32	215	Win 7 Pro SP1

VM HostName	CPU	Memory (Go)	Hard disk (Go)	OS
TUP (192.168.55.4)	4	20	100	CentOS x64 7.1

The software installed on the "HP ZBook" to run the VM is "Oracle Virtual Box 5.0.18"

The ITR VM is hosted by OVH cloud server.

Specifications for the ITR VM:

HostName	CPU	Memory (Go)	Hard disk (Go)	OS
vps319353.ovh.net (51.255.39.198)	1	2	10	Kubuntu 14.04

Software installed on VM "TUP":

- TranscityTM version 2.8.3 (Thales Ticketing Product).

Software installed on host "Transcity Back-Office :

- Oracle Java JDK 8
- TUP2ITR software component (Transcity TM to IT2Rail Gateway)
- Oracle Virtual Box

Software installed on VM "ITR":

- Oracle Java JDK 8
- ITR2TUP software component (IT2Rail Gateway to Transcity TM)

The schema below resumes the architecture of WP3-URBAN01 :

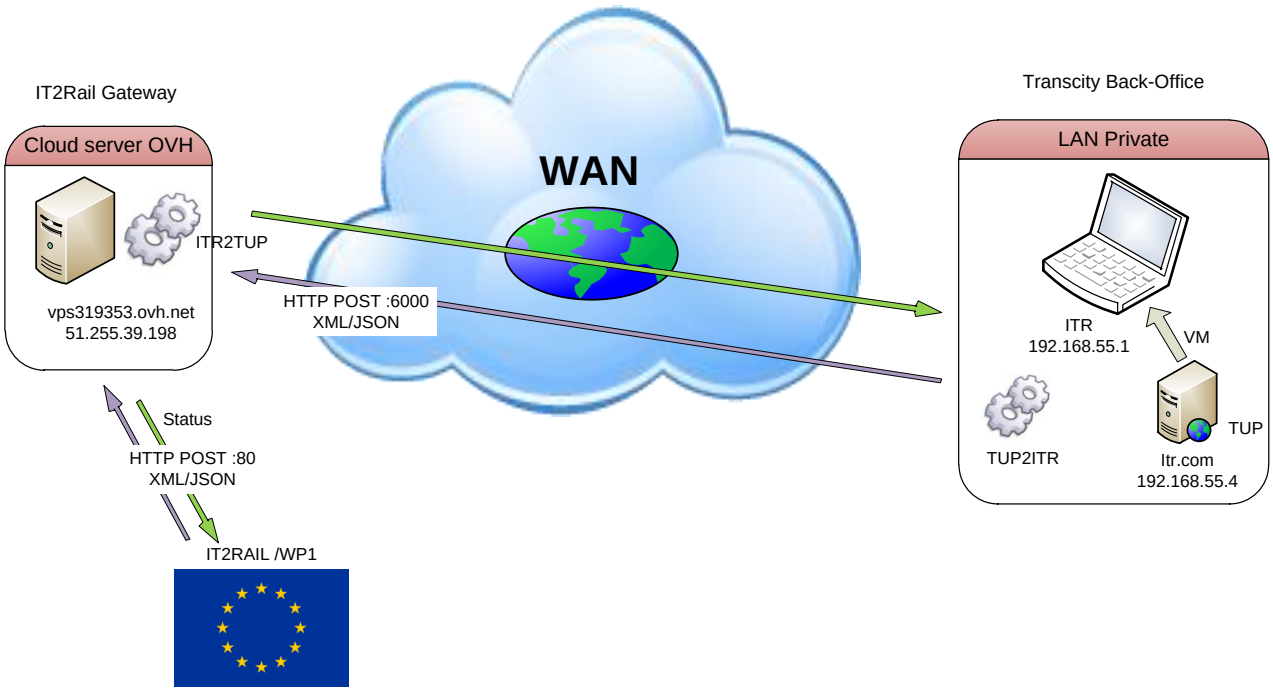


Figure 1: WP3-URBAN1 Infrastructure

4.5.2 Setup & Configuration

Here are listed the setup and configuration required to perform IT2Rail Final Release test campaign.

Start of VM.

Loading of topology

Loading of fare rules

Loading of dictionaries (operational parameter)

Load customer

Load business entities

Load Devices/Equipement

Load the “virtual” fare media

DNS Configuration

The PC “HP Zbook” hosting the TUP VM has DNS resolution on the following entries:

DNS Entry	Host-name
bo.it2r.com	TUP
m.it2r.com	TUP
services.it2r.com	TUP

Figure 2: URBAN DNS Configuration

The DNS resolution is done by editing the file C:/Windows/System32/drivers/etc.hosts

4.5.3 Tested System

The tested system is composed of:

- TranscityTM software version 2.8.3
- TUP2ITR & ITR2TUP software version 0.1

4.5.4 System Data Parameters

List here the data package to use in this test campaign. Example, for urban system stand-alone tests, it may include fare parameters, product catalog and topology parameters.

Ticketing Parameters are identified and provided as TICK-01

Fare Parameters are identified and provided as FARE-01

Product Catalog is identified and provided as PROD-01

Topology parameters are identified and provided as TOPO-01

Customer is identified and provided as CUSTOMER-01

Business Entities are identified and provided as BE-01

Equipement is identified and provided as DEVICE-01

Fare Media is identified and provided as MEDIA-01

Product Catalog

The Product Catalog PROD-01 is defined as follows:

	Metro	Bus
Multiride (5 travels)	24.50 euros	24.50 euros
Monthly pass	42.80 euros	42.80 euros

Figure 3: URBAN Product Catalog

Topology Parameters

The Topology Parameters TOPO-01 are defined as follows:

Line ID	Line	Open/Close	Transport Mode
1	Bus 1	CI Only	BUS

FarePoint ID	FarePoint Name
1	Schiphol, Valkweg
2	Schiphol, Airport
3	Nieuwe Meer, Koekoekslaan
4	Amsterdam, Anderlechtlaan
5	Amsterdam, Rijksmuseum
6	Amsterdam, Busstation Elandsgracht

Figure 4: URBAN unit test Topology

Customer

The Customer CUST-01 is defined as follows:

- Jane (jane@mail.com)
- Jena (jena@mail.com)
- Peter (peter@gmail.com)
- Steve (steve@gmail.com)

4.5.5 Simulators

Here below the simulators used for the IT2Rail test campaign are listed.

Simulated Equipment

The equipment simulated for IT2rail scenario are a subset of the full topology listed here:

Device ID	Device	Type	Line ID	Line	Transport Mode
51	VAL_51	VAL	5	Bus 1	BUS
68	VAL_68	VAL	6	Bus 2	BUS

Figure 5: URBAN Simulated Equipment

These devices are used to simulate the customer traveling experience through multiple urban legs (travel episodes)

These simulators are deployed and executed on the dedicated Thales infrastructure.

4.5.6 Personnel

There is no additional personnel required to perform this test campaign (apart from the Tester himself).

5. TEST DESCRIPTIONS

This chapter contains the test cases that are executed for the IT2Rail campaign.

5.1 AIR BOOKING AND ISSUANCE

5.1.1 Booking of an offer with air segment(s) and push-to-cloud message generation

WP3-AIRBOOK001	
Method Of Test	Demonstration
Type of test	Automated
Objectives	Booking air segments from an existing context.
Description	Invocation of the booking webservice that will trigger the booking of air segments from a context.
Status	NA
% passed	NA

WP3-AIRBOOK001	
Regression	NA
Test Case Tester	Amadeus

WP3-AIRBOOK001					
Id	**Step description**	**Expected result**	**Observed result**	**State**	**Associated defect**
	Preconditions: - internet connection with HTTP/HTTPS output to Amadeus web services authorised - A shopping context exists on Amadeus side: the recommendation contains two AF flights (AF1005Y/16NOV and AF1204Y/18NOV) - have SOAPUI or equivalent				
1	Invoke webservice TTR_ManageTripRQ 7.0 with a reference to the shopping context. The corresponding offer must contain at least one offer item with an air travel episode.	Itinerary offer with at least one air offer item is retrieved from shopping context for the itinerary offer booking. The booking details are then returned in the TTR_DisplayTripRS and in the push to cloud message.			

5.1.2 Issuance for booked offers with air segment(s) and push-to-cloud message generation

WP3-AIRISSUE001	
Method Of Test	Demonstration
Type of test	Automated
Objectives	This test will demonstrate the issuance of an Air entitlement.
Description	Invocation of the orchestration web service that will trigger the issuing of the air entitlement by forwarding the request to the air TSP.
Status	NA
% passed	NA

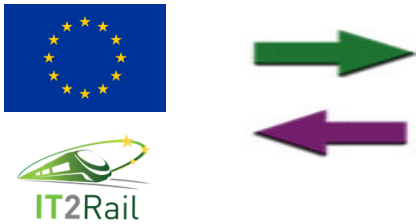
WP3-AIRISSUE001					
Regression	NA				
Test Case Tester	Amadeus				
Id	**Step description**	**Expected result**	**Observed result**	**State**	**Associated defect**
	Preconditions: - internet connection with HTTP/HTTPS output to Amadeus web services authorised - Perform booking and pricing of air segments prior to executing test - have SOAPUI or equivalent				

WP3-AIRISSUE001					
1	Invoke web service Ticket_OrchestrateDeals version 4.1 to perform the issuance for an existing booking containing at least one air segment	Result must be a successful Ticket_OrchestrateDeals response containing ticket number.			

5.2 URBAN BOOKING AND TICKETING

This category regroups test cases for issuing entitlement on URBAN TSP.

The 3 test cases correspond to the messages between IT2RAIL WP1 and WP3-URBAN1.



Operation	Type	URL	Received	Sent
SHOPPING (list of products)	GET	/shopping	✓	**Shopping.json**
BOOKING	POST	/booking	**Booking.json** - OfferItemtId	- PassengerId - ConfirmedBookingId
ENTITLEMENT	POST	/entitlement	**Entitlement.json** - PassengerId - ConfirmedBookingId	- EntitlementId - Tokens

Figure 6: WP3-URBAN1 communication

5.2.1 WP3-BT-SHOP-01: issuing product list

WP3-BT-ENTIT-01	
Method Of Test	Demonstration
Type of test	Manual (Semi-Automated)
Objectives	This test aims at demonstrating the ability of the ITR-GW component to achieve the issuance of a product list.
Description	Invocation of the “shopping” web service on ITR-GW component that will trigger the issuing of the urban ticketing product list by forwarding the request to the urban TSP (Transcity TM).
Status	NA
% passed	NA

Test Case 1: Issuing Entitlement – with default configuration WP3-URBAN01	
Test Case Running Status	NA
Test Case Tester	Edouard CARPENTIER

Id	Step description	Expected result	Observed result	State	Associated defect
	Preconditions: - The configuration WP3-URBAN01 is loaded - POSTMAN software is deployed on IT2Rail Test Platform				
1	Invoke web service “shopping” version 0.1 by executing POSTMAN with GET command	Result must be an HTTP response 200 OK with a JSON message (**see below**) containing the product list of the Transcity TM ticketing product)	-	**NA**	

The JSON message in expected result should match the following: {


```
  "catalog" : [ {
     "@type" : "MOBILE_PHONE_TICKET_ORDER",
    "catalogItemType" : {
      "value" : "MOBILE_PHONE_TICKET_ORDER",
      "label" : "Mobile phone ticket order"
    },
    "catalogItemId" : {
      "value" : 301,
      "label" : "Daily pass ticket on mobile phone"
    },
…
}, {
    "@type" : "ANONYMOUS_TICKET_ORDER",
    "catalogItemType" : {
      "value" : "ANONYMOUS_TICKET_ORDER",
      "label" : "Contactless ticket order"
    },
    "catalogItemId" : {
      "value" : 201,
      "label" : "Contactless ticket pre-loaded with a daily pass"
    },
…
```

5.2.2 WP3-BT-BOOK-01: Booking

WP3-BT-ENTIT-01	
Method Of Test	Demonstration
Type of test	Manual (Semi-Automated)
Objectives	This test aims at demonstrating the ability of the ITR-GW component to achieve the booking of an urban ticketing product.
Description	Invocation of the “entitlement” web service on ITR-GW component that will trigger the booking of the urban ticketing product by forwarding the request to the urban TSP (Transcity TM).
Status	NA
% passed	NA

Test Case 1: Issuing Entitlement – with default configuration WP3-URBAN01	
Test Case Running Status	NA
Test Case Tester	Edouard CARPENTIER

Id	Step description	Expected result	Observed result	State	Associated defect
	Preconditions: - The configuration WP3-URBAN01 is loaded - POSTMAN software is deployed on IT2Rail Test Platform				
1	Invoke web service “booking” version 0.1 by executing POSTMAN with provided script “WP3-BT-BOOK-01”	Result must be an HTTP response 200 OK with a JSON message (**see below**) containing “PassengerId” and the “BookingId”	-	**NA**	

Script **WP3-BT-BOOK-01:**


```
{
    "PassengerId": "2ed6e36b-06dc-5519-20a1-23b1c533e341",
    "BookingElements": {
        "BookingElement": {
            "BookingStatus": "complete",
            "ItineraryOfferItem": {
               "OfferItemProvider": "Thales",
               "OfferItemId": "201",
               "UnitsAvailable": "5",
               "SalesConditions": "https://www.thalesgroup.com/gts/rcs/salesconditions/item_423708/v2/en",
               "UnitsAvailable": "https://www.thalesgroup.com/gts/rcs/aftersalesconditions/item_423708/v2/en",
               "AfterSalesConditions": "https://www.thalesgroup.com/gts/rcs/aftersalesconditions/item_423708/v2/en",
               "OfferItemPriceId": "2ee6f36b-06ad-5919-23a1-23b1c444e341",
               "TravelEpisodeId": "2ed6e36b-06dc-5979-20a1-2211c508e341",
            }
        }
    }
}
```

The JSON message in expected result should match the following:


```
{
"PassengerId": "2ed6e36b-06dc-5519-20a1-23b1c533e341",
"BookingStatus": "ConfirmedBooking",
"BookingElements": {
    "BookingElement": {
        "BookingStatus": "complete",
        "ItineraryOfferItem": {
           "OfferItemProvider": "Thales",
           "OfferItemId": "201",
           "UnitsAvailable": "5",
           "SalesConditions": "https://www.thalesgroup.com/gts/rcs/salesconditions/item_423708/v2/en",
           "UnitsAvailable": "https://www.thalesgroup.com/gts/rcs/aftersalesconditions/item_423708/v2/en",
           "AfterSalesConditions": "https://www.thalesgroup.com/gts/rcs/aftersalesconditions/item_423708/v2/en",
           "AcceptedPaymentModes": {
                "PaymentMode": "SEPA",
                "PaymentMode": "VISA",
                "PaymentMode": "MASTERCARD",
                "PaymentMode": "PAYPAL"
           }
           "OfferItemPriceId": "2ee6f36b-06ad-5919-23a1-23b1c444e341",
           "TravelEpisodeId": "2ed6e36b-06dc-5979-20a1-2211c508e341",
        }
    }
},
"ConfirmedBooking": {
    "ConfirmedBookingId": "f1845654-b18a-c341-0ecf-c5c1c2eeeeaf",
    "ConfirmationMessage": "Your booking has been confirmed with reference number : f1845654-b18a-c341-0ecf-c5cf887acc",
    "IssuingStatus": "complete",
    "PaymentMeans_ProofOfSolveability": {
       "PaymentMean": {
            "Type": "EMV",
            "PaymentInformation": "hKwOWP62FEQIrYzlyiwqdBWg9Q7bQ7N9NczDg/DY6ZpBbUVudD1vIr9LAipYTTNBv7D7+W4D4gYjEbV",
            "Currency": "EUR",
```

```
            "Amount": "5.3"
        }
    }
}
}
```


5.2.3 WP3-BT-ENTIT-01: issuing entitlement and generate Tokens

WP3-BT-ENTIT-01	
Method Of Test	Demonstration
Type of test	Manual (Semi-Automated)
Objectives	This test aims at demonstrating the ability of the ITR-GW component to achieve the issuance of an urban entitlement.
Description	Invocation of the “entitlement” web service on ITR-GW component that will trigger the issuing of the urban entitlement and the tokens generation by forwarding the request to the urban TSP (Transcity TM).
Status	OK
% passed	100

Test Case 1 : Issuing Entitlement – with default configuration WP3-URBAN01	
Test Case Running Status	Run on 2016_08_23 at 11h10
Test Case Tester	Edouard CARPENTIER

Id	Step description	Expected result	Observed result	State	Associated defect
	Preconditions: **- The configuration WP3-URBAN01 is loaded** **- POSTMAN software is deployed on IT2Rail Test Platform**				
1	Invoke web service "entitlement" version 0.1 by executing POSTMAN with provided script "WP3-BT-ENT-01"	Result must be an HTTP response 200 OK with a JSON message (**see below**) containing "EntitlementId" and the generated tokens.	Returned JSON message is as expected (**see below**)	**OK**	

Script **WP3-BT-ENT-01** :

```
{
    "PassengerId": "2ed6e36b-06dc-5919-20a1-23b1c508e341",
    "ProviderIdentity": "hQEMA7EGdoXYdp6CAQf+IUpFbXYRJKHqYK4q4AX1sKjryZrzSyd7kurLPxXb1y2x",
    "OfferItemId": "2ee6e36b-06cd-5919-21a1-23b1c333e341",
    "ConfirmedBookingId": "f1869f54-b08a-c321-0ebf-c5c1c2fce6af"
}
```

The JSON message in expected result is : {


```
    "EntitlementId": "585129f4-d724-5502-4e0c-e88b03324b76",
    "PassengerId": "2ed6e36b-06dc-5919-20a1-23b1c508e341",
    "ProviderIdentity": "hQEMA7EGdoXYdp6CAQf+IUpFbXYRJKHqYK4q4AX1sKjryZrzSyd7kurLPxXb1y2x",
    "OfferItemId": "2ee6e36b-06cd-5919-21a1-23b1c333e341",
    "LegalInformation": "https://www.thalesgroup.com/gts/rcs/legal/000434/v1/en",
    "ContractReference": "https://www.thalesgroup.com/gts/rcs/contractreference/000434/v1/en",
    "SalesPolicies": "https://www.thalesgroup.com/gts/rcs/salespolicies/000434/v1/en",
    "Tokens": {
        "TokenId": "f1855f54-b58a-c331-0ebf-c5c1aaaae6af",
        "TokenId": "585129f4-d724-5502-4e0c-e88b03324b76"
    },
    "ConfirmedBookingId": "f1869f54-b08a-c321-0ebf-c5c1c2fce6af"
}
```

5.2.4 Booking of an offer with urban transport segment(s) and push-to-cloud message generation

WP3-URBANBOOK001	
Method Of Test	Demonstration
Type of test	Automated
Objectives	Booking urban transport segments from an existing context.
Description	Invocation of the booking webservice that will trigger the booking of urban transport segments from a context.
Status	NA
% passed	NA

WP3-URBANBOOK001					
Regression	NA				
Test Case Tester	Amadeus				
Id	**Step description**	**Expected result**	**Observed result**	**State**	**Associated defect**
	Preconditions: - internet connection with HTTP/HTTPS output to Amadeus web services authorised - A shopping context exists on Amadeus side: the recommendation contains at least one urban transport offer item - have SOAPUI or equivalent				

WP3-URBANBOOK001					
1	Invoke webservice TTR_ManageTripRQ 7.0 with a reference to the shopping context. The corresponding offer must contain at least one offer item with a urban transport travel episode.	Itinerary offer with at least one urban transport offer item is retrieved from shopping context for the itinerary offer booking. The booking details are then returned in the TTR_DisplayTripRS and in the push to cloud message.			

5.2.5 Issuance for booked offers with urban transport segment(s) and push-to-cloud message generation

WP3-URBANISSUE001	
Method Of Test	Demonstration
Type of test	Automated
Objectives	This test will demonstrate the issuance of an urban transport entitlement.
Description	Invocation of the orchestration web service that will trigger the issuing of the urban transport entitlement by forwarding the request to the air TSP.
Status	NA
% passed	NA

WP3-COACHISSUE001					
Regression	NA				
Test Case Tester	Amadeus				
Id	**Step description**	**Expected result**	**Observed result**	**State**	**Associated defect**
	Preconditions: - internet connection with HTTP/HTTPS output to Amadeus web services authorised - Perform booking of urban transport segments prior to executing test - have SOAPUI or equivalent				
1	Invoke web service Ticket_OrchestrateDeals version 4.1 to perform the issuance for an existing booking	Result must be a successful Ticket_OrchestrateDeals response containing ticket number.			

5.3 RAIL

This category regroups test for booking rail segment and issuing rail ticket….

5.3.1 WP3-RAIL-LOCK01

4.2.1 WP3-RAIL-LOCK01	
Method Of Test	Demonstration
Type of test	Automated
Objectives	Lock a RAIL inventory
Description	Invocation of the rail inventory lock web service to lock a seat on a train.
Status	NA
% passed	NA

WP3-RAIL-LOCK01	
Regression	NA
Test Case Tester	NA

Id	Step description	Expected result	Observed result	State	Associated defect
	Preconditions: - Internet connection and VPN with proper credentials - SOAPUI version 5.2.1 - INDRA01 data set as input in document "WP3_Booking_Ticketing_API_SOAP_v2.1", section "3.1.8 Request InventoryLock"				
1	Launch SOAPUI and execute InventoryLock01 XML Request	If itineraryOfferItem.UnitsAvailable>0, then Confirmation message that the lock has been performed. ….. <status>BLOCKED</status> ….. Otherwise ….. <status>PENDING</status>	NA	**NA**	NA

5.3.2 WP3-RAIL-GETPRICE01

4.2.1 WP3-RAIL-GETPRICE01	
Method Of Test	Demonstration
Type of test	Automated
Objectives	Get a price of itinerary Offer Item
Description	Invocation of web service to get the price of and offer item.
Status	NA
% passed	NA

WP3-RAIL- GETPRICE01	
Regression	NA
Test Case Tester	NA

Id	Step description	Expected result	Observed result	State	Associated defect
	Preconditions: - Internet connection and VPN with proper credentials - SOAPUI version 5.2.1 - GETPRICE01 data set as input in document “WP3_Booking_Ticketing_API_SOAP_v2.1”, section “3.2.2 Request GetPrice”				

Id	Step description	Expected result	Observed result	State	Associated defect
1	Launch SOAPUI and execute GetPrice XML Request	The merchant information and the provider signature of the item.	NA	**NA**	NA

5.3.3 WP3-RAIL-ISSUEENTITLEMENT01

4.2.1 WP3-RAIL- ISSUEENTITLEMENT01	
Method Of Test	Demonstration
Type of test	Automated
Objectives	Issue an entitlement
Description	Invocation of web service to issue an entitlement.
Status	NA
% passed	NA

WP3-RAIL-ISSUEENTITLEMENT01	
Regression	NA
Test Case Tester	NA

Id	Step description	Expected result	Observed result	State	Associated defect
	Preconditions: - Internet connection and VPN with proper credentials - SOAPUI version 5.2.1 - GETPRICE01 data set as input in document “WP3_Booking_Ticketing_API_SOAP_v2.1”, section “3.1.6 Request IssueEntitlement”				
1	Launch SOAPUI and execute ISSUEENTITLEMENT XML REQUEST	Response with a entitlement, contractReference, legacyInformation, passangerID, salesPolicies Tags information.	NA	**NA**	NA

5.3.4 WP3-RAIL-CONFIRMBOOKING01

4.2.1 WP3-RAIL-CONFIRMBOOKING01	
Method Of Test	Demonstration
Type of test	Automated
Objectives	Confirm a booking
Description	Invocation of web service to confirm a booking.
Status	NA
% passed	NA

WP3-RAIL-CONFIRMBOOKING01	
Regression	NA
Test Case Tester	NA

Id	Step description	Expected result	Observed result	State	Associated defect
	Preconditions: - Internet connection and VPN with proper credentials - SOAPUI version 5.2.1 - GETPRICE01 data set as input in document "WP3_Booking_Ticketing_API_SOAP_v2.1", section "3.1.4 Request ConfirmBooking"				
1	Launch SOAPUI and execute CONFIRMBOOKING XML REQUEST	Response with the confirmation. "The Booking has been made"	NA	**NA**	NA

5.3.5 WP3-RAIL-ISSUETOKEN01

4.2.1 WP3-RAIL-ISSUETOKEN01	
Method Of Test	Demonstration
Type of test	Automated
Objectives	Issue a token
Description	Invocation of web service to Issue a token.
Status	NA
% passed	NA

WP3-RAIL-ISSUETOKEN01	
Regression	NA
Test Case Tester	NA

Id	Step description	Expected result	Observed result	State	Associated defect
	Preconditions: - Internet connection and VPN with proper credentials - SOAPUI version 5.2.1 - GETPRICE01 data set as input in document "WP3_Booking_Ticketing_API_SOAP_v2.1", section "3.1.2 Request IssueToken"				

Id	Step description	Expected result	Observed result	State	Associated defect
1	Launch SOAPUI and execute ISSUETOKEN XML Request	Response with the Token id tags.	NA	**NA**	NA

5.3.6 Booking of an offer with rail segment(s) and push-to-cloud message generation

WP3-RAILBOOK001	
Method Of Test	Demonstration
Type of test	Automated
Objectives	Booking rail segments from an existing context.
Description	Invocation of the booking webservice that will trigger the booking of rail segments from a context.
Status	NA
% passed	NA

WP3-RAILBOOK001					
Regression	NA				
Test Case Tester	Amadeus				
Id	**Step description**	**Expected result**	**Observed result**	**State**	**Associated defect**
	Preconditions: - internet connection with HTTP/HTTPS output to Amadeus web services authorised - A shopping context exists on Amadeus side: the recommendation contains at least one rail offer item - have SOAPUI or equivalent				
1	Invoke webservice TTR_ManageTripRQ 7.0 with a reference to the shopping context. The corresponding offer must contain at least one offer item with a rail travel episode.	Itinerary offer with at least one rail offer item is retrieved from shopping context for the itinerary offer booking. The booking details are then returned in the TTR_DisplayTripRS and in the push to cloud message.			

5.3.7 Issuance for booked offers with rail segment(s) and push-to-cloud message generation

WP3-RAILISSUE001	
Method Of Test	Demonstration
Type of test	Automated
Objectives	This test will demonstrate the issuance of a Rail entitlement.
Description	Invocation of the orchestration web service that will trigger the issuing of the rail entitlement by forwarding the request to the air TSP.
Status	NA
% passed	NA

WP3-RAILISSUE001	
Regression	NA
Test Case Tester	Amadeus

Id	Step description	Expected result	Observed result	State	Associated defect
	Preconditions: - internet connection with HTTP/HTTPS output to Amadeus web services authorised - Perform booking of rail segments prior to executing test - have SOAPUI or equivalent				

WP3-RAILISSUE001					
1	Invoke web service Ticket_OrchestrateDeals version 4.1 to perform the issuance for an existing booking	Result must be a successful Ticket_OrchestrateDeals response containing ticket number.			

5.4 LONG DISTANCE BUSES

5.4.1 WP3-C01

WP3-C01	
Method Of Test	Demonstration
Type of test	Automated
Objectives	Shop
Description	Provide itinerary offer items
Status	NA
% passed	NA

WP3-LDB01	
Regression	NA
Test Case Tester	Filip Kvaček

Id	Step description	Expected result	Observed result	State	Associated defect
	Preconditions: - Internet connection				
1	Execute OLTIS-C01 script Search for segments within origin and destination with no service on specific date	Error message	-	**NA**	NA
2	Execute OLTIS-C01 script Search for segments within origin and destination with running service on specific date	List of available segments together with full fare amounts	-	**NA**	

5.4.2 WP3-C02

WP3-C02	
Method Of Test	Demonstration
Type of test	Automated
Objectives	Get detailed offer list
Description	Provide fare price
Status	NA
% passed	NA

WP3-LDB01	
Regression	NA
Test Case Tester	Filip Kvaček

Id	Step description	Expected result	Observed result	State	Associated defect
	Preconditions: - Internet connection				
1	Execute OLTIS-C02 script	Get detailed offer list	-	**NA**	NA

5.4.3 WP3-C03

WP3-C03	
Method Of Test	Demonstration
Type of test	Automated
Objectives	Booking/Lock an inventory
Description	Invocation of the inventory lock and block of seat
Status	NA
% passed	NA

WP3-LDB01	
Regression	NA
Test Case Tester	Filip Kvaček

Id	Step description	Expected result	Observed result	State	Associated defect
	Preconditions: - Internet connection				
1	Execute OLTIS-C03 script Lock an inventory	Confirmation reply with detail on fare and seat number	-	**NA**	NA

5.4.4 WP3-C04

WP3-C04	
Method Of Test	Demonstration
Type of test	Automated
Objectives	Issue ticket
Description	Deliver Entitlement/Token/Embodiment
Status	NA
% passed	NA

WP3-LDB01	
Regression	NA
Test Case Tester	Filip Kvaček

Id	Step description	Expected result	Observed result	State	Associated defect
	Preconditions: - Internet connection				
1	Execute OLTIS-C04 script Process Entitlement/Token/Embodiment	Confirmation reply with contract reference	-	**NA**	NA

5.4.5 Booking of an offer with coach segment(s) and push-to-cloud message generation

WP3-COACHBOOK001	
Method Of Test	Demonstration
Type of test	Automated
Objectives	Booking coach segments from an existing context.
Description	Invocation of the booking webservice that will trigger the booking of coach segments from a context.
Status	NA
% passed	NA

WP3-COACHBOOK001					
Regression	NA				
Test Case Tester	Amadeus				
Id	**Step description**	**Expected result**	**Observed result**	**State**	**Associated defect**
	Preconditions: - internet connection with HTTP/HTTPS output to Amadeus web services authorised - A shopping context exists on Amadeus side: the recommendation contains at least one coach offer item - have SOAPUI or equivalent				

WP3-COACHBOOK001					
1	Invoke webservice TTR_ManageTripRQ 7.0 with a reference to the shopping context. The corresponding offer must contain at least one offer item with a coach travel episode.	Itinerary offer with at least one coach offer item is retrieved from shopping context for the itinerary offer booking. The booking details are then returned in the TTR_DisplayTripRS and in the push to cloud message.			

5.4.6 Issuance for booked offers with coach segment(s) and push-to-cloud message generation

WP3-COACHISSUE001	
Method Of Test	Demonstration
Type of test	Automated
Objectives	This test will demonstrate the issuance of a coach entitlement.
Description	Invocation of the orchestration web service that will trigger the issuing of the coach entitlement by forwarding the request to the air TSP.
Status	NA
% passed	NA

WP3-COACHISSUE001					
Regression	NA				
Test Case Tester	Amadeus				
Id	**Step description**	**Expected result**	**Observed result**	**State**	**Associated defect**
	Preconditions: - internet connection with HTTP/HTTPS output to Amadeus web services authorised - Perform booking of coach segments prior to executing test - have SOAPUI or equivalent				
1	Invoke web service Ticket_OrchestrateDeals version 4.1 to perform the issuance for an existing booking	Result must be a successful Ticket_OrchestrateDeals response containing ticket number.			

6. TEST EXECUTION

The following chapter presents the different test results of the previously defined test cases. Following heads of chapter are the identification of the test runs.

6.1 CORE-AIR-01

6.1.1 Booking of an offer with air segment(s) and push-to-cloud message generation

WP3-AIRBOOK001	
Method Of Test	Demonstration
Type of test	Automated
Objectives	Booking air segments from an existing context.
Description	Invocation of the booking webservice that will trigger the booking of air segments from a context.
Status	Passed
% passed	100

WP3-AIRBOOK001	
Regression	NA
Test Case Tester	Amadeus

Id	Step description	Expected result	Observed result	State	Associated defect
	Preconditions: - internet connection with HTTP/HTTPS output to Amadeus web services authorised - A shopping context exists on Amadeus side: the recommendation contains two AF flights (AF1005Y/16NOV and AF1204Y/18NOV) - have SOAPUI or equivalent				
1	Invoke webservice TTR_ManageTripRQ 7.0 with a reference to the shopping context. The corresponding offer must contain at least one offer item with an air travel episode.	Itinerary offer with at least one offer item is retrieved from shopping context for the itinerary offer booking. The booking details are then returned in the TTR_DisplayTripRS and in the push to cloud message.		**OK**	Booking messages  TTR_DisplayTripRS.rtf Push-to-cloud message:  PushToCloudAir.rtf

6.1.2 Issuance for booked offers with air segment(s) and push-to-cloud message generation

WP3-AIRISSUE001	
Method Of Test	Demonstration
Type of test	Automated
Objectives	This test will demonstrate the issuance of an Air entitlement.
Description	Invocation of the orchestration web service that will trigger the issuing of the air entitlement by forwarding the request to the air TSP.
Status	NA
% passed	NA

WP3-AIRISSUE001					
Regression	NA				
Test Case Tester	Amadeus				
Id	**Step description**	**Expected result**	**Observed result**	**State**	**Associated defect**
	Preconditions: - internet connection with HTTP/HTTPS output to Amadeus web services authorised - Perform booking and pricing of air segments prior to executing test - have SOAPUI or equivalent				

WP3-AIRISSUE001					
1	Invoke web service Ticket_OrchestrateDeals version 4.1 to perform the issuance for an existing booking containing at least one air segment	Result must be a successful Ticket_OrchestrateDeals response containing ticket number.	The issuance is performed successfully	**OK**	Issuance messages:  Issuance_air.xml Push-to-cloud message:  Push-to-cloud-issuance-air.xml

6.2.1 WP3-RAIL-LOCK01

4.2.1 WP3-RAIL-LOCK01	
Method Of Test	Demonstration
Type of test	Automated
Objectives	Lock a RAIL inventory
Description	Invocation of the rail inventory lock web service to lock a set on a train.
Status	OK
% passed	100

WP3-RAIL-LOCK01	
Regression	NA
Test Case Tester	NA

Id	Step description	Expected result	Observed result	State	Associated defect
	Preconditions: - Internet connection and VPN with proper credentials - SOAPUI version 5.2.1 - INDRA01 data set as input in document "WP3_Booking_Ticketing_API_SOAP_v2.1", section "3.1.8 Request InventoryLock"				
1	Launch SOAPUI and execute InventoryLock XML Request script	If itineraryOfferItem.UnitsAvailable>0, then Confirmation message that the lock has been performed. ….. <status>BLOCKED</status> ….. Otherwise ….. <status>PENDING</status> …..	itineraryOfferItem.UnitsAvailable=15 result: <status>BLOCKED</status> itineraryOfferItem.UnitsAvailable=0 result: <status>PENDING</status>	**OK**	NA

- XML result itineraryOfferItem.UnitsAvailable=15

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns2:InventoryLockResponse xmlns:ns2="http://indra.es.transporte">
      <BookingElementsList>
        <status>BLOCKED</status>
        <itineraryOfferItem>
          <salesConditions/>
          <afterSalesConditions/>
          <paymentMode>VISA</paymentMode>
          <travelEpisodeID>
            <travelEpisodeID>EPID4567</travelEpisodeID>
          </travelEpisodeID>
          <offerItemPrice/>
          <offerItemProvider>OIP987</offerItemProvider>
          <unitsAvailable>15</unitsAvailable>
        </itineraryOfferItem>
      </BookingElementsList>
    </ns2:InventoryLockResponse>
  </soap:Body>
</soap:Envelope>
```

- XML result itineraryOfferItem.UnitsAvailable=0

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns2:InventoryLockResponse xmlns:ns2="http://indra.es.transporte">
      <BookingElementsList>
        <status>PENDING</status>
        <itineraryOfferItem>
          <salesConditions/>
          <afterSalesConditions/>
          <paymentMode>VISA</paymentMode>
          <travelEpisodeID>
            <travelEpisodeID>EPID4567</travelEpisodeID>
          </travelEpisodeID>
          <offerItemPrice/>
          <offerItemProvider>OIP987</offerItemProvider>
          <unitsAvailable>0</unitsAvailable>
        </itineraryOfferItem>
      </BookingElementsList>
    </ns2:InventoryLockResponse>
  </soap:Body>
</soap:Envelope>
```

6.2.2 WP3-RAIL-GETPRICE01

4.2.1 WP3-RAIL-GETPRICE01	
Method Of Test	Demonstration
Type of test	Automated
Objectives	Get a price of itinerary Offer Item
Description	Invocation of web service to get the price of and offer item.
Status	OK
% passed	100

WP3-RAIL- GETPRICE01	
Regression	NA
Test Case Tester	NA

Id	Step description	Expected result	Observed result	State	Associated defect
	Preconditions: - Internet connection and VPN with proper credentials - SOAPUI version 5.2.1 - GETPRICE01 data set as input in document "WP3_Booking_Ticketing_API_SOAP_v2.1", section "3.2.2 Request GetPrice"				
1	Launch SOAPUI and execute GETPRICE XML Request script	The merchant information and the provider signature of the item.	<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"> <soap:Body> <ns2:GetPriceResponse xmlns:ns2="http://indra.es.transporte"> <OfferItemPriceItem> <merchantInformation>Merchant Information</merchantInformation> <providerSignature>JDUER74UJRKFO98483UEKEORLOEKLTIKT849</providerSignature> </OfferItemPriceItem> </ns2:GetPriceResponse> </soap:Body> </soap:Envelope>	**OK**	NA

6.2.3 WP3-RAIL-ISSUEENTITLEMENT01

4.2.1 WP3-RAIL- ISSUEENTITLEMENT01	
Method Of Test	Demonstration
Type of test	Automated
Objectives	Issue an entitlement
Description	Invocation of web service to issue an entitlement.
Status	OK
% passed	100

WP3-RAIL-ISSUEENTITLEMENT01	
Regression	NA
Test Case Tester	NA

Id	Step description	Expected result	Observed result	State	Associated defect
	Preconditions: - Internet connection and VPN with proper credentials - SOAPUI version 5.2.1 - GETPRICE01 data set as input in document "WP3_Booking_Ticketing_API_SOAP_v2.1", section "3.1.6 Request IssueEntitlement"				

1	Launch SOAPUI and execute ISSUEENTITLEMENT XML REQUEST script	Response with an entitlement, contractReference, legacyInformation, passangerID, salesPolicies Tags information.	<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"> <soap:Body> <ns2:IssueEntitlementResponse xmlns:ns2="http://indra.es.transporte"> <Entitlement> <tokenIDList> <ID>TID0002</ID> </tokenIDList> <tokenIDList xsi:nil="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/> <contractReference>REF00001</contractReference> <legacyInformation>legacyInformation</legacyInformation> <passangerID>792706958</passangerID> <salesPolicies>2 days</salesPolicies> </Entitlement> </ns2:IssueEntitlementResponse> </soap:Body> </soap:Envelope>	**OK**	NA

6.2.4 WP3-RAIL-CONFIRMBOOKING01

4.2.1 WP3-RAIL-CONFIRMBOOKING01	
Method Of Test	Demonstration
Type of test	Automated
Objectives	Confirm a booking
Description	Invocation of web service to confirm a booking.
Status	OK
% passed	100

WP3-RAIL-CONFIRMBOOKING01	
Regression	NA
Test Case Tester	NA

Id	Step description	Expected result	Observed result	State	Associated defect
	Preconditions: - Internet connection and VPN with proper credentials - SOAPUI version 5.2.1 - GETPRICE01 data set as input in document "WP3_Booking_Ticketing_API_SOAP_v 2.1", section "3.1.4 Request ConfirmBooking"				
1	Launch SOAPUI and execute CONFIRMBOOKING XML REQUEST script	Response with the confirmation. "The Booking has been made"	<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"> <soap:Body> <ns2:ConfirmBookingResponse xmlns:ns2="http://indra.es.transporte"> <ConfirmedBooking> <issuingStatus>**ISSUED**</issuingStatus> <confirmationMessage>The Booking has been made</confirmationMessage> </ConfirmedBooking> </ns2:ConfirmBookingResponse> </soap:Body> </soap:Envelope>	**OK**	NA

6.2.5 WP3-RAIL-ISSUETOKEN01

4.2.1 WP3-RAIL-ISSUETOKEN01	
Method Of Test	Demonstration
Type of test	Automated
Objectives	Issue a token
Description	Invocation of web service to Issue a token.
Status	OK
% passed	100

WP3-RAIL-ISSUETOKEN01	
Regression	NA
Test Case Tester	NA

Id	Step description	Expected result	Observed result	State	Associated defect
	Preconditions: - Internet connection and VPN with proper credentials - SOAPUI version 5.2.1 - GETPRICE01 data set as input in document "WP3_Booking_Ticketing_API_SOAP_v2.1", section "3.1.2 Request IssueToken"				

Id	Step description	Expected result	Observed result	State	Associated defect
1	Launch SOAPUI and execute ISSUETOKEN XML Request script	Response with the Token id tags.	*Refer to the XML snippet below*	**OK**	NA

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns2:IssueTokenResponse xmlns:ns2="http://indra.es.transporte">
      <TokenList>
        <tokenID>
          <ID>1472033495697</ID>
        </tokenID>
        <entitlementID>
          <ID>1472033495698</ID>
        </entitlementID>
        <tappingModuleDescription>
          <tappingModuleVersion/>
          <tappingModuleRequirements/>
          <name/>
          <activityName/>
          <downloadURI/>
        </tappingModuleDescription>
      </TokenList>
      <TokenList>
        <tokenID>
          <ID>1472033495697</ID>
        </tokenID>
        <entitlementID>
          <ID>1472033495698</ID>
```

```
        </entitlementID>
        <tappingModuleDescription>
          <tappingModuleVersion/>
          <tappingModuleRequirements/>
          <name/>
          <activityName/>
          <downloadURI/>
        </tappingModuleDescription>
      </TokenList>
    </ns2:IssueTokenResponse>
  </soap:Body>
</soap:Envelope>
```

6.2.6 Booking of an offer with rail segment(s) and push-to-cloud message generation

WP3-RAILBOOK001	
Method Of Test	Demonstration
Type of test	Automated
Objectives	Booking rail segments from an existing context.
Description	Invocation of the booking webservice that will trigger the booking of rail segments from a context.
Status	NA
% passed	NA

WP3-RAILBOOK001					
Regression	NA				
Test Case Tester	Amadeus				
Id	**Step description**	**Expected result**	**Observed result**	**State**	**Associated defect**
	Preconditions: - internet connection with HTTP/HTTPS output to Amadeus web services authorised - A shopping context exists on Amadeus side: the recommendation contains at least one rail offer item - have SOAPUI or equivalent				

WP3-RAILBOOK001					
1	Invoke webservice TTR_ManageTripRQ 7.0 with a reference to the shopping context. The corresponding offer must contain at least one offer item with a rail travel episode.	Itinerary offer with at least one rail offer item is retrieved from shopping context for the itinerary offer booking. The booking details are then returned in the TTR_DisplayTripRS and in the push to cloud message.	The itinerary offer is successfully booked	**OK**	Broker message:  BrokerRS_Train.rtf Booking response:  TTR_DisplayTripRS_Train.rtf Push-to-cloud message:  PushToCloudTrain.rtf

6.2.7 Issuance for booked offers with rail segment(s) and push-to-cloud message generation

WP3-RAILISSUE001	
Method Of Test	Demonstration
Type of test	Automated
Objectives	This test will demonstrate the issuance of a Rail entitlement.
Description	Invocation of the orchestration web service that will trigger the issuing of the rail entitlement by forwarding the request to the air TSP.
Status	NA
% passed	NA

WP3-RAILISSUE001					
Regression	NA				
Test Case Tester	Amadeus				
Id	**Step description**	**Expected result**	**Observed result**	**State**	**Associated defect**
	Preconditions: - internet connection with HTTP/HTTPS output to Amadeus web services authorised - Perform booking of rail segments prior to executing test - have SOAPUI or equivalent				

WP3-RAILISSUE001					
1	Invoke web service Ticket_OrchestrateDeals version 4.1 to perform the issuance for an existing booking	Result must be a successful Ticket_OrchestrateDeals response containing ticket number.	The issuance is performed successfully	**OK**	Issuance messages: Issuance_rail.xml Push-to-cloud message: Push-to-cloud-issuance-rail.xml

6.3.1 WP3-C01

WP3-C01	
Method Of Test	Demonstration
Type of test	Automated
Objectives	Shop
Description	Provide itinerary offer items
Status	OK

WP3-LDB01	
Regression	NA
Test Case Tester	Filip Kvaček

Id	Step description	Expected result	Observed result	State	Associated defect
	Preconditions: - Internet connection				
1	Execute OLTIS-C01 script Search for segments within origin and destination with no service on specific date	Error message	{"Message":"No connection found.","Code":10108}	**OK**	NA

Id	Step description	Expected result	Observed result	State	Associated defect
2	Execute OLTIS-C01 script Search for segments within origin and destination with running service on specific date	List of available segments together with full fare amounts	{"handle":1875803050,"totalConnections":2,"connections":[{"line":"000088","conn":9,"fromTime":"2017-10-20T18:00:00","fromStop":"Praha [*CZ],,ÚAN Florenc","fromStopCode":27862,"stand":"4","toTime":"2017-10-20T22:15:00","toStop":"Berlin [*D],,ZOB am Funkturm","toStopCode":999,"km":363,"price":{"eur":2260},"priceRet":{"eur":4083},"flags":65545,"startSale":"1970-01-01T00:00:00","carrier":"EUROLINES"},{"line":"000279","conn":7,"fromTime":"2017-10-20T18:45:00","fromStop":"Praha [*CZ],,ÚAN Florenc","fromStopCode":27862,"stand":"3","toTime":"2017-10-20T23:15:00","toStop":"Berlin [*D],,ZOB am Funkturm","toStopCode":999,"km":-1,"price":{"eur":1422},"flags":44,"startSale":"1970-01-01T00:00:00","carrier":"Bohemian Lines","freePlaces":8}],"mpz":"GB","currency":1}	**Passed**	NA

6.3.2 WP3-C02

WP3-C02	
Method Of Test	Demonstration
Type of test	Automated
Objectives	Get detailed offer list
Description	Provide fare price
Status	OK

WP3-LDB01	
Regression	NA
Test Case Tester	Filip Kvaček

Id	Step description	Expected result	Observed result	State	Associated defect
	Preconditions: - Internet connection				

1	Execute OLTIS-C02 script	Get detailed offer list	{"prices":[{"tarCode":"P","fareType":"full fare","eur":2260,"kc":62000,"bgn":4800},{"tarCode":"S26","fareType":"student to 26 years","eur":2034,"kc":55800,"bgn":4300},{"tarCode":"DU","fareType":"senior over 60 years","eur":2034,"kc":55800,"bgn":4300},{"tarCode":"D4C","fareType":"children 4-12 years","eur":1130,"kc":31000,"bgn":2400},{"tarCode":"DT4","fareType":"children to 4 years","eur":452,"kc":12400,"bgn":1000}],"pricesRet":[{"tarCode":"P","fareType":"two-way full fare","eur":4083,"kc":112000,"bgn":8700},{"tarCode":"S26","fareType":"two-way student to 26 years","eur":3675,"kc":100800,"bgn":7800},{"tarCode":"DU","fareType":"two-way senior over 60 years","eur":3675,"kc":100800,"bgn":7800},{"tarCode":"D4C","fareType":"two-way children 4-12 years","eur":2042,"kc":56000,"bgn":4300},{"tarCode":"DT4","fareType":"two-way children 4years","eur":817,"kc":22400,"bgn":1700}],"busses":[{"busNumber":"1","startNum":9,"busFlags":1,"places":[{"num":1,"row":4},{"num":2,"row":3},{"num":3,"row":1},{"num":4},{"num":5,"row":4,"col":1},{"num":6,"row":3,"col":1},{"num":7,"row":1,"col":1},{"num":8,"col":1},{"num":9,"row":4,"col":2},{"num":10,"row":3,"col":2},{"num":11,"row":1,"col":2,"f":4},{"num":12,"col":2,"f":4},{"num":13,"row":4,"col":3,"f":4},{"num":14,"row":3,"col":3,"f":4},{"num":15,"row":1,"col":3,"f":4},{"num":16,"col":3,"f":4},{"num":17,"row":4,"col":5,"f":4},{"num":18,"row":3,"col":5,"f":4},{"num":19,"row":1,"col":5,"f":4},{"num":20,"col":5,"f":4},{"num":21,"row":4,"col":6,"f":4},{"num":22,"row":3,"col":6,"f":4},{"num":23,"row":1,"col":6,"f":4},{"num":24,"col":6,"f":4},{"num":25,"row":4,"col":7,"f":4},{"num":26,"row":3,"col":7,"f":4},{"num":27,"row":1,"col":7,"f":4},{"num":28,"col":7,"f":4},{"num":29,"row":4,"col":8,"f":4},{"num":30,"row":3,"col":8,"f":4},{"num":31,"row":1,"col":8,"f":4},{"num":32,"col":8,"f":4},{"num":33,"row":4,"col":9,"f":4},{"num":34,"row":3,"col":9,"f":4},{"num":35,"row":1,"col":9,"f":4},{"num":36,"col":9,"f":4},{"num":37,"row":4,"col":10,"f":4},{"num":38,"row":	**OK**	NA

			3,"col":10,"f":4},{"num":39,"row":2,"col":10,"f":4},{"num":40,"row":1,"col":10,"f":4},{"num":41,"col":10,"f":4}]}]}		

6.3.3 WP3-C03

WP3-C03	
Method Of Test	Demonstration
Type of test	Automated
Objectives	Booking/Lock an inventory
Description	Invocation of the inventory lock and block of seat
Status	OK

WP3-LDB01	
Regression	NA
Test Case Tester	Filip Kvaček

Id	Step description	Expected result	Observed result	State	Associated defect
	Preconditions: - Internet connection				
1	Execute OLTIS-C03 script Lock an inventory	Confirmation reply with detail on fare and seat number	{"ticketHandle":" 6f2d06fa-13bd-4a64-a496-39f95ba374f2","tickets":[{"direction":0,"busThere":"1","price":{"tarCode":"P","fareType":"full fare","eur":2260},"required":10}]	**OK**	NA

6.3.4 WP3-C04

WP3-C04	
Method Of Test	Demonstration
Type of test	Automated
Objectives	Issue ticket
Description	Deliver Entitlement/Token/Embodiment
Status	OK

WP3-LDB01	
Regression	NA
Test Case Tester	Filip Kvaček

Id	Step description	Expected result	Observed result	State	Associated defect
	Preconditions: - Internet connection				
1	Execute OLTIS-C04 script Process Entitlement/Token/Embodiment	Confirmation reply with contract reference	[{"transCode":"MF0XGM","price":{"eur":2260},"pdfUri":"http://eshopcv.amsbus.cz/idos/kosik/PDF/?p=MF0XGMQslpVo%2bNRvEVGTN2fOC%2bcg%3d%3d"}]	**OK**	NA

6.3.5 Booking of an offer with coach segment(s) and push-to-cloud message generation

WP3-COACHBOOK001	
Method Of Test	Demonstration
Type of test	Automated
Objectives	Booking coach segments from an existing context.
Description	Invocation of the booking webservice that will trigger the booking of coach segments from a context.
Status	NA
% passed	NA

WP3-COACHBOOK001					
Regression	NA				
Test Case Tester	Amadeus				
Id	**Step description**	**Expected result**	**Observed result**	**State**	**Associated defect**
	Preconditions: - internet connection with HTTP/HTTPS output to Amadeus web services authorised - A shopping context exists on Amadeus side: the recommendation contains at least one coach offer item - have SOAPUI or equivalent				

WP3-COACHBOOK001					
1	Invoke webservice TTR_ManageTripRQ 7.0 with a reference to the shopping context. The corresponding offer must contain at least one offer item with a coach travel episode.	Itinerary offer with at least one coach offer item is retrieved from shopping context for the itinerary offer booking. The booking details are then returned in the TTR_DisplayTripRS and in the push to cloud message.	The itinerary offer is successfully booked	**OK**	Broker message: BrokerRS_Coach.rtf Booking response: TTR_DisplayTripRS_Coach.rtf Push-to-cloud message: PushToCloudCoach.rtf

6.3.6 Issuance for booked offers with coach segment(s) and push-to-cloud message generation

WP3-COACHISSUE001	
Method Of Test	Demonstration
Type of test	Automated
Objectives	This test will demonstrate the issuance of a coach entitlement.
Description	Invocation of the orchestration web service that will trigger the issuing of the coach entitlement by forwarding the request to the air TSP.
Status	NA
% passed	NA

WP3-COACHISSUE001					
Regression	NA				
Test Case Tester	Amadeus				
Id	**Step description**	**Expected result**	**Observed result**	**State**	**Associated defect**
	Preconditions: - internet connection with HTTP/HTTPS output to Amadeus web services authorised - Perform booking of coach segments prior to executing test - have SOAPUI or equivalent				

WP3-COACHISSUE001					
1	Invoke web service Ticket_OrchestrateDeals version 4.1 to perform the issuance for an existing booking	Result must be a successful Ticket_OrchestrateDeals response containing ticket number.	The issuance is performed successfully	**OK**	Issuance messages:  Issuance_coach.xml Push-to-cloud message:  Push-to-cloud-issuance-coach.xml

This campaign was run on the 23rd of August 2016.

6.4.1 WP3-BT-ENTIT-01: issuing entitlement

WP3-BT-ENTIT-01	
Method Of Test	Demonstration
Type of test	Manual (Semi-Automated)
Objectives	This test aims at demonstrating the ability of the ITR-GW component to achieve the issuance of an urban entitlement.
Description	Invocation of the “entitlement” web service on ITR-GW component that will trigger the issuing of the urban entitlement by forwarding the request to the urban TSP (Transcity TM).
Status	OK
% passed	100

Test Case 1: Issuing Entitlement – with default configuration WP3-URBAN01	
Regression	NA
Test Case Tester	Thales

Test Case 1: Issuing Entitlement – with default configuration WP3-URBAN01					
Id	**Step description**	**Expected result**	**Observed result**	**State**	**Associated defect**
	Preconditions: - The configuration WP3-URBAN01 is loaded - POSTMAN software is deployed on IT2Rail Test Platform				
1	Invoke web service “entitlement” version 0.1 by executing POSTMAN with provided script “WP3-BT-ENT-01”	Result must be an HTTP response 200 OK with a JSON message (**see below**) containing “EntitlementId” that will be incremented by 1 each time this test is re-executed (matching an ItemOrderId in the Transcity TM ticketing product)	Returned JSON message is as expected (**see below**)	**OK**	

The JSON message in expected result is: { **"EntitlementId": "000000022-01"**, "LegalInformation": "https://www.thalesgroup.com/gts/rcs/legal/000434/v1/en", "ContractReference": "https://www.thalesgroup.com/gts/rcs/contractreference/000434/v1/en", "SalesPolicies": "https://www.thalesgroup.com/gts/rcs/salespolicies/000434/v1/en", "ProviderIdentity": "hQEMA7EGdoXYdp6CAQf+IUpFbXYRJKHqYK4q4AX1sKjryZrzSyd7kurLPxXb1y2x", "PassengerId": "2ed6e36b-06dc-5919-20a1-23b1c508e341", "ItineraryOfferItemId": "2ee6e36b-06cd-5919-21a1-23b1c333e341", "Tokens": { "TokenId": "f1855f54-b58a-c331-0ebf-c5c1aaaae6af", "TokenId": "585129f4-d724-5502-4e0c-e88b03324b76" }, "ConfirmedBookingId": "f1845654-b18a-c341-0ecf-c5c1c2eeeeaf"}

6.4.2 WP3-BT-ENTIT-02: issuing tokens

WP3-BT-ENTIT-02	
Method Of Test	Demonstration
Type of test	Manual (Semi-Automated)
Objectives	This test will demonstrate the issuance of an urban token corresponding to an entitlement.
Description	Invocation of the issuance web service on ITR-GW component that will trigger the issuing of the urban token by forwarding the request to the urban TSP (Transcity TM).
Status	KO
% passed	50

Test Case 1: Issuing token as a JSON message "TokenId" – with default configuration WP3-URBAN01					
Regression	NA				
Test Case Tester	Thales				
Id	**Step description**	**Expected result**	**Observed result**	**State**	**Associated defect**
	Preconditions: - The configuration WP3-URBAN01 is loaded - POSTMAN software is deployed on IT2Rail Test Platform				
1	Invoke web service "token" version 0.1 by executing POSTMAN with provided script "WP3-BT-ENT-02"	HTTP response 200 OK with a JSON message containing "TokenId" that will be unique per test campaign (**see below**)	The result was a HTTP 404 fault page.	**Passed**	

The JSON message in expected result is: { **"TokenId": " f1855f54-b58a-c331-0ebf-c5c1aaaae6af " }**

Test Case 2: Issuing token as a QRCode image – with default configuration WP3-URBAN01					
Regression	NA				
Test Case Tester	Thales				
Id	**Step description**	**Expected result**	**Observed result**	**State**	**Associated defect**
	Preconditions: - The configuration WP3-URBAN01 is loaded - POSTMAN software is deployed on IT2Rail Test Platform				
1	Invoke web service “token_qrcode” version 0.1 by executing POSTMAN with provided script “WP3-BT-ENT-02”	Result must be an image PNG file representing a QRCode (**see below - 1**)	The returned result is a JSON message with “TokenId”	**Passed**	
2	Scan the received QRCode image with any standard QRCode Reader Android application	You should read the content of the QRCode as a JSON message with “TokenId” (**see below - 2**)	The content for the QRCode is an ASCII String containing a GUID but not a JSON message	**Failed**	

6.4.3 WP3-BT-FARE-01: Compute fare price

WP3-BT-FARE-01	
Method Of Test	Demonstration
Type of test	Manual (Semi-Automated)
Objectives	This test will demonstrate the pricing of an OfferItem.
Description	Invocation of the issuance web service on ITR-GW component that will trigger the pricing of the offer item in parameter by forwarding the request to the urban TSP (Transcity TM).
Status	OK
% passed	100

Test Case 1: Computing Fare Price – with default configuration WP3-URBAN01	
Regression	NA
Test Case Tester	Thales

Id	Step description	Expected result	Observed result	State	Associated defect
	Preconditions: - The configuration WP3-URBAN01 is loaded - POSTMAN software is deployed on IT2Rail Test Platform				
1	Invoke web service “price” version 0.1 by executing POSTMAN with provided script “WP3-BT-FARE-01”	Result must be an HTTP response 200 OK with a JSON message containing an “OfferItem Price” with an amount of 24.50 euros (**see below**)	Result is as expected	**Passed**	

The JSON message in expected result is: { **"ItineraryOfferItemId": "3ee6e36b-06cd-5919-21a1-23b1c333e34"**, "**ItineraryOfferItemPrice**": "**24.50 EUR** }

6.4.4 Booking of an offer with urban transport segment(s) and push-to-cloud message generation

WP3-URBANBOOK001	
Method Of Test	Demonstration
Type of test	Automated
Objectives	Booking urban transport segments from an existing context.
Description	Invocation of the booking webservice that will trigger the booking of urban transport segments from a context.
Status	NA
% passed	NA

WP3-URBANBOOK001	
Regression	NA
Test Case Tester	Amadeus

WP3-URBANBOOK001					
Id	**Step description**	**Expected result**	**Observed result**	**State**	**Associated defect**
	Preconditions: - internet connection with HTTP/HTTPS output to Amadeus web services authorised - A shopping context exists on Amadeus side: the recommendation contains at least one urban transport offer item - have SOAPUI or equivalent				
1	Invoke webservice TTR_ManageTripRQ 7.0 with a reference to the shopping context. The corresponding offer must contain at least one offer item with a urban transport travel episode.	Itinerary offer with at least one urban transport offer item is retrieved from shopping context for the itinerary offer booking. The booking details are then returned in the TTR_DisplayTripRS and in the push to cloud message.	The itinerary offer is successfully booked	**OK**	

6.4.5 Issuance for booked offers with urban transport segment(s) and push-to-cloud message generation

WP3-URBANISSUE001	
Method Of Test	Demonstration
Type of test	Automated
Objectives	This test will demonstrate the issuance of an urban transport entitlement.
Description	Invocation of the orchestration web service that will trigger the issuing of the urban transport entitlement by forwarding the request to the air TSP.

WP3-URBANISSUE001	
Status	NA
% passed	NA

WP3-COACHISSUE001					
Regression	NA				
Test Case Tester	Amadeus				
Id	**Step description**	**Expected result**	**Observed result**	**State**	**Associated defect**
	Preconditions: - internet connection with HTTP/HTTPS output to Amadeus web services authorised - Perform booking of urban transport segments prior to executing test - have SOAPUI or equivalent				
1	Invoke web service Ticket_OrchestrateDeals version 4.1 to perform the issuance for an existing booking	Result must be a successful Ticket_OrchestrateDeals response containing ticket number.	The issuance is performed successfully	**OK**	

This campaign was run on the 30th of August 2016.

6.5.1 WP3-BT-ENTIT-02: issuing tokens

WP3-BT-ENTIT-02	
Method Of Test	Demonstration
Type of test	Manual (Semi-Automated)
Objectives	This test will demonstrate the issuance of an urban token corresponding to an entitlement.
Description	Invocation of the issuance web service on ITR-GW component that will trigger the issuing of the urban token by forwarding the request to the urban TSP (Transcity TM).
Status	OK
% passed	100

Test Case 1: Issuing token as a JSON message “TokenId” – with default configuration WP3-URBAN01	
Regression	NA
Test Case Tester	Thales

Test Case 1: Issuing token as a JSON message “TokenId” – with default configuration WP3-URBAN01					
Id	**Step description**	**Expected result**	**Observed result**	**State**	**Associated defect**
	Preconditions: - The configuration WP3-URBAN01 is loaded - POSTMAN software is deployed on IT2Rail Test Platform				
1	Invoke web service “token” version 0.1 by executing POSTMAN with provided script “WP3-BT-ENT-02”	HTTP response 200 OK with a JSON message containing “TokenId” that will be unique per test campaign (**see below**)	The returned result is a JSON message with “TokenId”	**OK**	

The JSON message in expected result is: {**"TokenId":"f1855f54-b58a-c331-0ebf-c5c1aaaae6af"}**

Test Case 2: Issuing token as a QRCode image – with default configuration WP3-URBAN01	
Regression	NA
Test Case Tester	Thales

Id	Step description	Expected result	Observed result	State	Associated defect
	Preconditions: - The configuration WP3-URBAN01 is loaded - POSTMAN software is deployed on IT2Rail Test Platform				
1	Invoke web service “token_qrcode” version 0.1 by executing POSTMAN with provided script “WP3-BT-ENT-02”	Result must be an image PNG file representing a QRCode (**see below - 1**)	The returned result is a JSON message with “TokenId”	**OK**	
2	Scan the received QRCode image with any standard QRCode Reader Android application	You should read the content of the QRCode as a JSON message with “TokenId” (**see below - 2**)	The content for the QRCode is an ASCII String containing a GUID but not a JSON message	**OK**	

1 - The scanned QRCode “reveals” a JSON message that is like this: {**"TokenId":"h1855f54-b58a-c331-0ebf-c5c1aaaae6af "}**

2 - The QRCode PNG image is the following: