Deliverable D2.2

System Pre-Integration

Grant agreement
No 951725

5GRAIL

5G for future RAILway mobile communication system

Deliverable D2.2 - Toba Integration report

Due date of deliverable: 30/11/2021

Actual submission date: 31/01/2023

Leader/Responsible of this Deliverable: Kontron

Reviewed: Y

Document status		
Revision	**Date**	**Description**
0.1	01/12/2021	First issue
0.2	25/01/2022	Document updated according to V2 of D2.1 interface and with editorial updates
0.3	08/04/2022	Additional updates and consolidation of the first deliverable version
1.0	22/04/2022	Last version of D2.2 integrating comments from Consortium. Version submitted
1.1	14/12/2022	Conclusions chapter added to the document
1.2	10/01/2023	Revised version following latest EC comments
2.0	27/01/2023	Final version after consortium review – version submitted

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: 01/11/2020 Duration: 30 months

Executive Summary

The purpose of the D2.2 System Pre-Integration deliverable is to describe the test cases that have been performed prior to the lab test activities (WP3 and WP4). These pre-integration tests will ease the lab test integration by validating the logical interface between the applications and the FRMCS gateway. Each application supplier will perform the test cases described in this document and will report a feedback that will be included in the document.

These tests will be performed locally, by using simulators. Therefore, any information about the setup or configuration of the 5G bearer or network will not be relevant for this deliverable (this information is part of the lab and/or field test reports).

This is the version 1.0 of the document, that includes the description of the test cases that need to be excecuted, the description of the two gateway simulators, as well as the test execution results.

Abbreviations and Acronyms

Abbreviation	Description
API	Application Programmable Interface
ATP	Automatic Train Protection
ATO	Automatic Train Operation
CAF	Construcciones y Auxiliar de Ferrocarriles
CCTV	Closed Circuit TeleVision
CPU	Central Processing Unit
ETCS	European Train Control System
FRMCS	Future Railway Mobile Communication System
FQDN	Fully Qualified Domain Name
GTW or GW	GaTeWay or GateWay
IMS	IP Multimedia Subsystem
IP	Internet Protocol
JSON	JavaScript Object Notation
MCx	Mission Critical
OB	On Board
OB_GTW	On-Board Gateway
OBA	On-Board Application (e.g. ETCS on-board, ATO on-board)
OBU	On-Board Unit
PIS	Passenger Information System
RAM	Random Access Memory
RBC	Remote Block Centre
REST	REpresentational State Transfer
RPC	Remote Procedure Call
TCMS	Train Control Management System
TCP	Transmission Control Protocol
TLS	Transport Layer Security

TS	Track Side
TS_GTW	TrackSide Gateway
UIC	Union Internationale des Chemins de fer
VPN	Virtual Private Network
WP2	Work Package 2
WS	WebSocket
WSS	WebSocketSecure

CONTENTS

Table of figures

List of tables

1 Introduction

The main objective of the 5GRail project is to integrate the railway applications with the FRMCS system in order to benefit from the features and innovation provided by the 5G technology.

The pre-integration phase will provide a framework that will enhance the 5G technology integration in the railway environment in terms of risks and cost. The objective of the system pre-integration is to enable a step-by-step integration in lab, in order to minimize the risks of possible OBapp/TSapp interface issues.

The pre-integration phase will test ATO, ETCS, TCMS, PIS, CCTV Offload/CCTV Live and voice applications, that will be sharing the 5G resources in the future railway environment.

The aim is not to validate the application functional test cases (which will be performed in WP3/WP4 5G labs), that is why cross border scenarios are out of scope of this document. The focus is to test the protocol stack and messages exchanged between the apps and the FRMCS GW. Therefore it is not necessary for this phase to have a complete solution of the applications or FRMCS GW.

In order to be more effective in the overall delivery of the project and as these tests could be taken as a preliminary step for the tests which will be carried out in WP3 and WP4, the system pre-integration will be made in the laboratories defined on those work packages.

The following chapters contain information about how the FRCMS Gateway and each application will develop the pre-integration and the test plan.

2 On-Board and Track Side Gateways simulators

2.1 Introduction

The FRMCS ecosystem consists of a symmetric architecture including on-board and trackside gateways. The objective of this chapter is to describe the different simulators that will be used for the validation of the test cases defined in chapter 3.

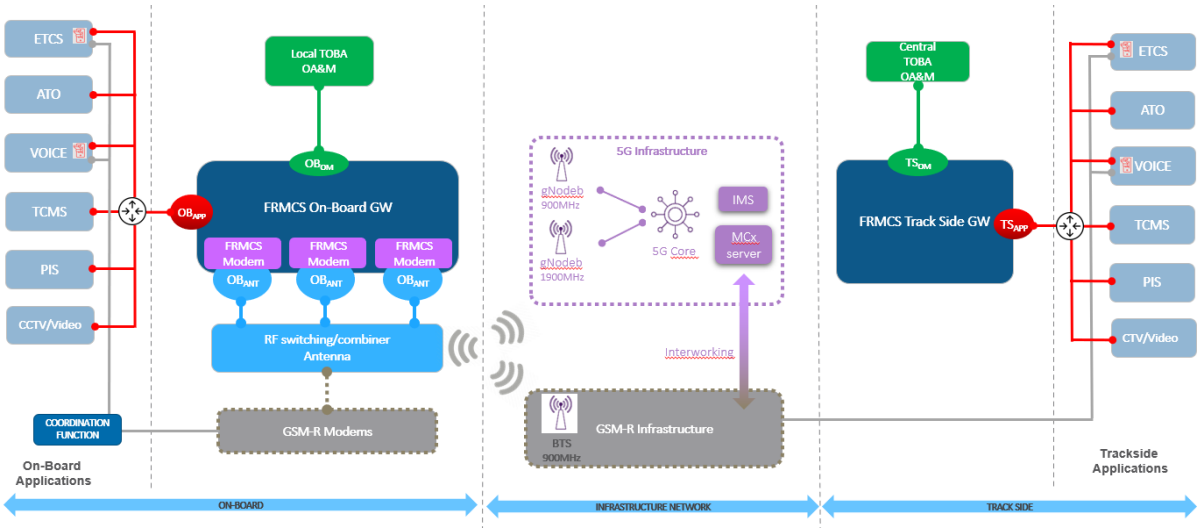

Figure 1: Architecture overview (Component's list)

The pre-integration is expected to be tested in two phases:

1. The first phase consists of the validation of the OBapp interface locally. Alstom will provide a simulator based in python that will allow application suppliers to test this interface.
2. The second phase consists of validating the OBapp/TSapp interfaces by performing an end-to-end communication. Kontron will provide a simulator that will allow application suppliers to test the end-to-end communication in a locally simulated environment (no special HW required).

This chapter contains a brief description of the simulator for each phase, as well as an explanation of the steps to be followed in order to use the simulators.

Note: Alstom has also provided an enhanced version of the Phase 1 simulator with both OBapp and TSapp interfaces to test also the incoming session in auto_accept mode. See chapter 2.2.1.2 for a more detailed description.

2.2 Phase 1: OBApp Interface Simulator (Alstom)

2.2.1 Architecture and hardware/software detail

2.2.1.1 Simulator Phase 1 – version 1.x

A Python prototype of OB_{APP} client and server is provided for testing purposes. This allows to simulate both sides of the OB_{APP} interface (client and server) and check the OB_{APP} API methods.

This prototype is made of two Python programs:

- obapp_server.py: a server for OB_{APP} API (to simulate the OB_GTW)
- obapp_client.py: a client for OB_{APP} API (to simulate an application)

Besides, a requirement file (named requirements.txt) is attached and contains the required Python modules to be installed.

2.2.1.2 Simulator Phase 1 – version 2.x

An enhanced version of simulator was provided and consists in a virtual machine with two network interfaces to simulate OB_{APP} server on one side and TS_{APP} server on the other one. This allows the triggering of incoming session messages in auto_accept mode only.

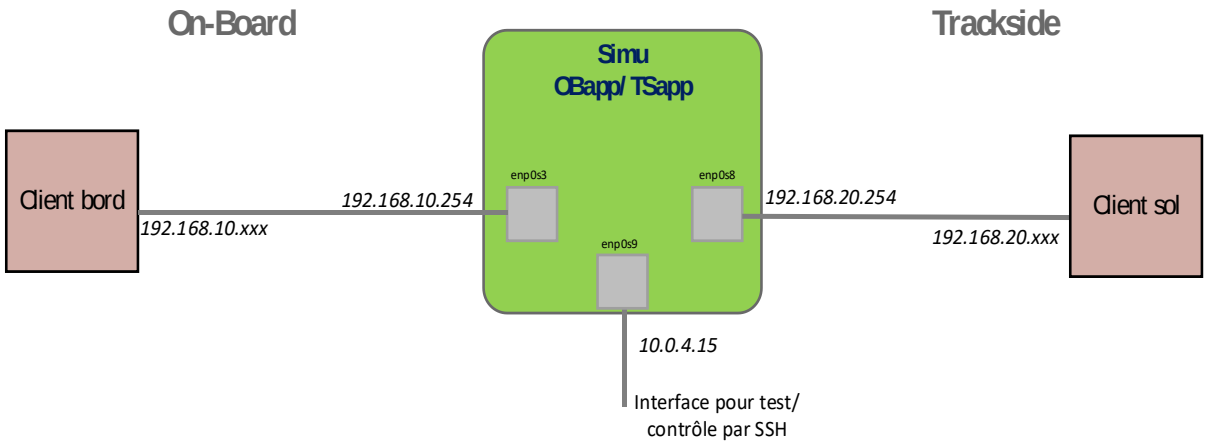

Figure 2 - Simulator Phase 1 - version 2

Note: a third network interface (enp0s9) is also given for practical purposes.

2.2.2 Release note

Few versions have been provided for this simulator, as described in the table below:

Version	Date of release	Specification reference
Simu_Ph1.v1.1	18/10/2021	D2.1 rev 3
Simu_Ph1.v1.2	17/11/2021	D2.1 rev 3.1 (draft)
Simu_Ph1.v2.1	16/02/2022	D2.1 rev 3.2 (draft) Include two interfaces to simulate incoming sessions.

The last version of OBAPP interface simulator is dated November 17th 2021.

The functions available in this simulator are compliant with the OB_{APP} API definition in chapter 6 of D2.1 document, revision 3.

Nevertheless, some restrictions are to be considered:

1. There is no TLS in these prototypes (ws instead of wss. See RFC 6455 for detailed information.)
2. Notifications: only SESSION_STATUS_CHANGED implemented in v1, following a change of status. In v2, INCOMING_SESSION_NOTIF is also implemented.
3. For SERVICE_REQUEST function, only the connection_status can be requested (no gnss positioning request)
4. In v1, all the "incoming_auto" modes are possible, but they are not used by the server since there is no mechanism to trigger an incoming session. In v2, only auto_accept and auto_reject modes are implemented.

2.2.3 Configuration and execution guide for Phase 1 - v1.x

2.2.3.1 Recommendation

A requirements file which contains the required python modules to be installed. Use `pip install -r requirements.txt` or install these modules manually. Make sure these modules are reachable by the program itself (in the PYTHONPATH)

It is recommended to use Python3 with a version <3.9. There are some deprecation issues with latest Python version (3.10).

Client and server can be run on the same device (then they communicate using localhost), or two different devices.

2.2.3.2 Use the server

Open a command prompt and go to the directory containing the python server file (using `cd` command). It is possible to use a Python virtual environment to make it run but the full procedure is not described in this document.

Then, launch the server:

```
python obapp_server.py
```

Note: For this prototype, the server is listening on a non-standard TCP port (8765). However, in a « real » implementation, it would be with TLS on port 443.

To connect to the server from a client on a different device, if the IP address of the server is 192.168.1.1, then the WebSocket URI to be used is: **ws://192.168.1.1:8765/v0**

It is important to have the version (v0) at the end of the URI, it is checked by the server for the WebSocket opening.

2.2.3.3 Use the client

Open a second command prompt (if you have already opened the server on a first command prompt) and go to the directory containing the python server file (using `cd` command).

Then, launch the client:

```
python obapp_client.py
```

To connect to an OB_{APP} server:

1. If the OB_{APP} server is the Python server described above, and if it runs on the same device, just use the command `connect` on the client.
2. If the OB_{APP} server runs on a different device with IP address 192.168.1.1 for example, run the command `connect ws://192.168.1.1:8765/v0`

Once the WebSocket is established, you can run the interesting functions on the client side.

Type 'help' to see the possible commands:

```
(OBapp) help

Documented commands (type help <topic>):
========================================
connect     disconnect  quit  register     session_start   specific_services
deregister  help        raw   session_end  session_status

(OBapp)
```

Type 'help <*function*>' to see a slight description and the parameters to be set for the function:

```
(OBapp) help register

        Register an OBapp application

        register <application_type> <originator_id> <mode> <incoming_auto>

        - application_type:     Type of application (integer [1..7])
        - originator_id:        FQDN of the application (string)
        - mode:                 Coupling mode the application
                                (string: "loose" or "tight")
        - incoming_auto:        Accept or reject incoming session
                                (string: "auto_accept" or "auto_reject"
                                or "not_auto")
```

2.2.3.4 Examples

We consider here that you have launched the client and connect to an OB_{APP} server (the Python simulator or another one). The following command are performed on the Python client.

1. Make a registration:

```
(OBapp) register 1 my_id loose auto_accept
>>> {"method": "register", "id": 1, "params": {"application_type": 1, "originator_id": "my_id", "mode": "loose", "incoming_auto": "
auto_accept"}, "jsonrpc": "2.0"}
<<< {"result": {"app_uuid": "f25c5f98-8ac5-46fe-b639-46ba40a1e4aa"}, "id": 1, "jsonrpc": "2.0"}
```

We receive the app_uuid to be used in the following steps

2. Start a session to a TS host:

```
(OBapp) session_start bbbd9771-0b0e-4a3d-a2bb-fa08b8c9f246 google.fr 6 7911 10
>>> {"method": "session_start", "id": 1, "params": {"app_uuid": "bbbd9771-0b0e-4a3d-a2bb-fa08b8c9f246", "addr": "google.
fr", "protocol": 6, "port": 7911, "comm_profile": 10}, "jsonrpc": "2.0"}
<<< {"result": {"session_uuid": "ba95135f-9b7e-4670-adf8-6aceb3600ceb", "ip_dest": "172.16.1.1"}, "id": 1, "jsonrpc": "2
.0"}
(OBapp) <<< notification {"method": "session_status_changed", "params": {"session_uuid": "ba95135f-9b7e-4670-adf8-6aceb3
600ceb", "session_status": "trying"}, "jsonrpc": "2.0"}
<<< notification {"method": "session_status_changed", "params": {"session_uuid": "ba95135f-9b7e-4670-adf8-6aceb3600ceb",
 "session_status": "working"}, "jsonrpc": "2.0"}
```

We receive the answer with a session_uuid and the remote IP address to be used for the user plane.

We receive notifications for status change (a first one with “Trying” state, and a second one with “Working” state).

Note: if you are using the Python OB_{APP} server, the Working state occurs 3 seconds after the Trying state (it includes a temporization to simulate a time to reach the real TS equipment), if the address sent in the SESSION_START request is known (make a try with google.fr for example).

1. Run a SESSION_STATUS on this new session:

```
(OBapp) session_status bbbd9771-0b0e-4a3d-a2bb-fa08b8c9f246 ba95135f-9b7e-4670-adf8-6aceb3600ceb
>>> {"method": "session_status", "id": 2, "params": {"app_uuid": "bbbd9771-0b0e-4a3d-a2bb-fa08b8c9f246", "session_uuid":
 "ba95135f-9b7e-4670-adf8-6aceb3600ceb"}, "jsonrpc": "2.0"}
<<< {"result": {"status": "working"}, "id": 2, "jsonrpc": "2.0"}
```

2. Close the session with SESSION_END:

```
<<< notification {"method": "session_status_changed", "params": {"session_uuid": "ba95135f-9b7e-4670-adf8-6aceb3600ceb",
 "session_status": "deleted"}, "jsonrpc": "2.0"}
<<< {"result": {}, "id": 3, "jsonrpc": "2.0"}
```

We receive the SESSIONS_STATUS_CHANGED notification (to “deleted” status) and the answer of the request, which is empty.

2.2.4 Configuration and execution guide for Phase 1 - v2.x

2.2.4.1 Recommendation

The Simu Ph1.v2.1 is a VM (.ova file) which must be imported in VirtualBox.

The client shall be configured with the following IP address:

- **192.168.10.xxx** for OB clients (not 254 which is for the simu)
- **192.168.20.xxx** for TS clients (not 254 which is for the simu)

2.2.4.2 Use the server

Step 1: import Simu.ova in VirtualBox.

Step 2: launch the VM, use the following credentials:

Login	Password
kindy	passuser
su	passroot

Step 3: check if the network configuration is adapted to your use case (see slide 5). Then run the script of network setting named simu-ip-config.sh (you should be in root to do that, command: su):

```
kindy@simu:~$ su
Mot de passe : ********
root@simu:/home/kindy# ./simu-ip-config.sh
```

Step 4: quit root mode (mandatory) and launch Python script for Obapp/Tsapp server:

```
root@simu:/home/kindy# exit
kindy@simu:~$ python3.8 obapp_server.py
```

Some deprecation message may appear but do not consider it.

Then, the simulator is listening for WebSocket connections on two interfaces “OB” and “TS”.

By default, enp0s3 and enp0s8 are in « internal network » mode. There are two ways to test OB and TS clients with the simu:

- If the clients are also a VM in the same host machine, do not change the setting of the simu. Name of the virtual network: ineth1 for OB, ineth2 for TS
- If the clients are on a separated machine, change the network mode of the VM accordingly.

Websocket URI:

- URI to open a Websocket from the OB side: **ws://192.168.10.254:8765/v0**
- URI to open a Websocket from the TS side: **ws://192.168.20.254:8765/v0**

2.3 Phase 2: OBApp and TSApp Simulator (Kontron)

2.3.1 Architecture and hardware/software detail

Phase 2 simulator aims to test further the OBapp and TSapp interfaces once Phase 1 is achieved.

Indeed, Phase 2 will simulate the OBapp and TSapp API with a robot that will treat the message flow and feedback according to the interface specification.

Concretely, the Phase 2 simulator will be a Virtual Machine with 4 external IPs addressed (1 for OBapp, 1 for TSapp, 1 OB user-plane, 1 for TS user plane) as depicted below.

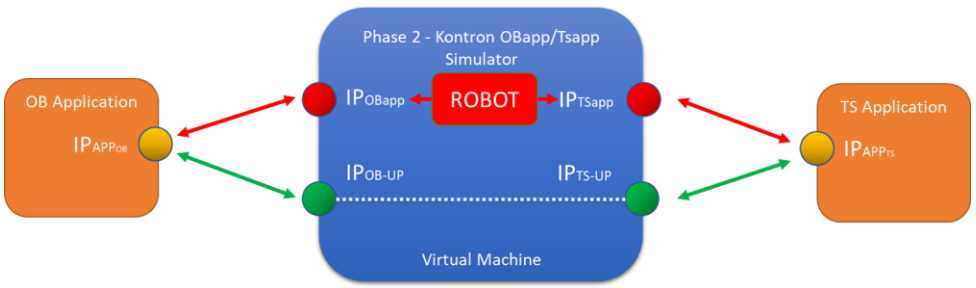

Figure 3: Phase 2 – Kontron OBapp/TSapp Simulator design/concept.

The simulator capabilities:

1. It is only for loose coupled application
2. It will support the full OBapp/TSapp specification as specified in D2.1 V2.
3. it will support 1 OB application + 1 TS application
4. it will support only 1 session between these 2 applications

2.3.2 Configuration and execution guide

2.3.2.1 Configuration

The VM has to be installed on a PC. The PC should have following configuration:

5. The Host OS is Ubuntu server 20.04 LTS.
6. The KVM hypervisor should be available on the PC.
7. Minimal configuration should be: 2 CPU, 4 GB RAM, a 50GB disk.
8. The PC needs to have one Ethernet port dedicated to both the OB and the TS interfaces.
9. 3 Ethernet cards: one for management, one for OB GTW and the last one for TS GTW.

To be able to debug, it should be mandatory to have a nethawk spy on the OB and TS interfaces to capture exchanged messages.

2.3.2.2 Execution guide

The OB application and TS application will use the OB/TSapp IP addresses and interact similarly with nominal OB and TS FRMCS gateway.

The websocket establishment is done between the OB and the TS applications and the robot. Then, the following messages are processed by the robot:

1. The FRMCS_GTW_REGISTER: this message is straight away acknowledged by the robot. No check is done on the parameter values.
2. The FRMC_GTW_DEREGISTER: the message is acknowleded by the robot except if the application uuid is not found. In that case, it is negatively acknowledged.
3. The FRMCS_GTW_SESSION_START: for this message, the robot's behaviour is:
    1. The message is acknowledged. No check is done on the parameter values.
    2. An FRMCS_APP_SESSION_STATUS_CHANGED(trying) message is sent to the application by the robot.
    3. The robot then sends an FRMCS_APP_INCOMING_SESSION_REQ or an FRMCS_APP_INCOMING_SESSION_NOTIF message to the counterpart application, depending on the value of the incoming_auto field received in the FRMCS_GTW_REGISTER message of this counterpart application.
    4. The robot sends a FRMCS_APP_STATUS_CHANGED(deleted) message to the initiator application if the destinator incoming_auto parameter is set to auto_reject.
    5. The robot sends an FRMCS_APP_STATUS_CHANGED(working) message if the incoming_auto is set to auto_accept.
4. The FRMCS_GTW_INCOMING_SESSION_RSP:
    1. The robot sends an FRMCS_APP_SESSION_STATUS_CHANGED(working) message to both application if the return_code receipt in the message is OK.

2. The robot sends an FRMCS_APP_SESSION_STATUS_CHANGED(deleted) message to both application if the return_code receipt in the message is NOT_OK.

5. The FRMCS_GTW_SESSION_END:
    1. The message is acknowledged by the robot. No check is done on the parameter values.
    2. An FRMCS_APP_SESSION_STATUS_CHANGED(deleted) is sent to the counterpart application.
6. The FRMCS_GTW_SESSION_STATUS: the robot sents the current session status to the application. No check is done on the parameter values.
7. The FRMCS_GTW_SERVICE_REQUEST:
    1. Only the connection_status value of the request_type is taken into account by the

    robot.
    2. The robot sends the current connection_status accordingly to the definition described in the D2.1 - Architecture Report (§6.2.4.4.6).

In this version of the robot, only nominal cases are managed. No validity check of the different field values are done, except for the uuid ones.

3 Test Cases

3.1 Introduction

The tests described below are focused on the OBApp interface between the onboard simulators and FRMCS Onboard Gateway. However, please note that same principles are applied for TSApp and FRMCS Trackside Gateway and same tests will be done for this environment for loose coupled applications.

As these pre-integration tests aim to validate the OBApp/TSApp interface from the functional point of view, the TLS protocol specified in the OBApp/TSApp interface will not be implemented nor tested in the scope of D2.2. It is understood that TLS protocol is a standard protocol that could add more complexity to the pre-integration tests and would not provide added value to them.

It is important to note also, that some test cases are only available in the phase 2 simulator. Those test cases include a note in the description specifying such limitation. By default, all test cases can be tested by both simulators with equal result.

3.2 Loose Coupled Applications (Alstom, CAF, Teleste, Thales)

3.2.1 TC_001: Registration of an application with FRMCS Gateway

3.2.1.1 Purpose

The purpose of this test is to validate the ability to register an application in the FRMCS network.

3.2.1.2 Description of initial state/configuration

The initial state covers the following steps:

- Application onboard equipment are installed and configured
- FRMCS Onboard Gateway is connected and configured to application equipment
- Application equipment are connected and power on in nominal state

3.2.1.3 Test procedure: Successful registration of an application

STEP	ACTION	EXPECTED RESULT(S)
1	Open a WebSocket connection with the gateway.	The connection is established successfully.
2	FRMCS_GTW_REGISTER Request JSON message is sent to FRMCS Gateway with the specified values. (See next tables).	- FRMCS_GTW_REGISTER Response JSON message is received with the specified values. (See next tables). - Application with its uuid is registered in the FRMCS network.

Table 1 TC_001 Test procedure

FRMCS_GTW_REGISTER Request Json Structure

FIELD	TYPE	VALUE
jsonrpc	String	2.0
method	String	register
id	String	Implementation dependant
params	Object	See next table.

Table 2 TC_001 FRMCS_GTW_REGISTER Request JsonRPC parameters

FIELD	TYPE	VALUE
application_type	Int	4 (using TCMS as an example)
originator_id	String	ID of the host For example with FQDN format: *Ex :* *<ETCS ID>.<ETCS ID type>*.etcs.frmcs
mode	String	Loose
incoming_auto	String	auto_accept/auto_reject/not_auto (depending on the application)

Table 3 TC_001 FRMCS_GTW_REGISTER Request parameters

FRMCS_GTW_REGISTER Response parameters

FIELD	TYPE	VALUE
jsonrpc	String	2.0
id	String	same value as in the FRMCS_GTW_REGISTER request
result	Object	See next table.

Table 4 TC_001 FRMCS_GTW_REGISTER Response JsonRPC parameters

FIELD	TYPE	VALUE
app_uuid	String	A new local ID unique chosen by itself if the request is succeeded. Must be build following RFC4122

Table 5 TC_001 FRMCS_GTW_REGISTER Response parameters

3.2.1.4 Success criteria

- All partial results are as expected.
- No error messages

3.2.2 TC_002: Deregistration of an application with FRMCS Gateway

3.2.2.1 Purpose

The purpose of this test is to validate the ability to deregister an application in the FRMCS network.

3.2.2.2 Description of initial state/configuration

The initial state covers the following steps:

- Application onboard equipment are installed and configured
- FRMCS Onboard Gateway is connected and configured to application equipment
- Application equipment are connected and power on in nominal state
- Application is simulated to be registered in the FRMCS network via FRCMS Onboard Gateway
- Websocket already established and registration already performed

3.2.2.3 Test procedure: Successful deregistration of an application

STEP	ACTION	EXPECTED RESULT(S)
1	FRMCS_GTW_DEREGISTER Request JSON message is sent to FRMCS Gateway with the specified values. (See next tables).	- FRMCS_GTW_DEREGISTER Response JSON message is received with the specified values. (See next tables). - Application with its uuid is deregistered from the FRMCS network

Table 6 TC_002 Test procedure

FRMCS_GTW_DEREGISTER Request parameters

FIELD	TYPE	VALUE
jsonrpc	String	2.0
method	String	deregister
id	String	Implementation dependant
params	Object	See next table.

Table 7 TC_002 FRMCS_GTW_DEREGISTER Request JsonRPC parameters

FIELD	TYPE	VALUE
app_uuid	String	Uuid of the application

Table 8 TC_002 FRMCS_GTW_DEREGISTER Request parameters

FRMCS_GTW_DEREGISTER Response parameters

FIELD	TYPE	VALUE
jsonrpc	String	2.0
id	String	same value as in the FRMCS_GTW_DEREGISTER request
result	Object	Empty

Table 9 TC_002 FRMCS_GTW_DEREGISTER Response JsonRPC parameters

FIELD	TYPE	VALUE
N/A	N/A	N/A

Table 10 TC_002 FRMCS_GTW_DEREGISTER Response parameters

3.2.2.4 Success criteria

- All partial results are as expected
- No error messages.

3.2.3 TC_003: Session establishment request of an application with FRMCS onboard Gateway

3.2.3.1 Purpose

The purpose of this test is to validate the ability to launch a session establishment request in the FRMCS network.

3.2.3.2 Description of initial state/configuration

The initial state covers the following steps:

- Application onboard equipment are installed and configured
- FRMCS Onboard Gateway is connected and configured to application equipment
- Application equipment are connected and power on in nominal state
- The application is simulated to be registered in the FRMCS network via FRCMS Onboard Gateway.
- Websocket already established and registration already performed

3.2.3.3 Test procedure: Successful establishment of an application session

STEP	ACTION	EXPECTED RESULT(S)
1	FRMCS_GTW_SESSION_START Request JSON message is sent to FRMCS Gateway with the specified values. (See next tables).	- FRMCS_GTW_SESSION_START Response JSON message is received with the specified values. (See next tables). - Two consecutive FRMCS_APP_SESSION_STATUS_CHANGED notification JSON message are received with the specified values. (See next tables). - The new session is registered in the FRMCS network with its uuid.

Table 11 TC_003 Test procedure

FRMCS_GTW_SESSION_START Request Json Structure

FIELD	TYPE	VALUE
jsonrpc	String	2.0
method	String	session_start
id	String	Implementation dependant
params	Object	See next table.

Table 12 TC_003 FRMCS_GTW_SESSION_START Request JsonRPC parameters

FIELD	TYPE	VALUE
app_uuid	String	A preconfigured simulated application uuid.
addr	String	IP address XXX.XXX.XXX
protocol	Int	6 (TCP)
port_dest	Int	XXXX preconfigured port.
comm_profile	String	1

Table 13 TC_003 FRMCS_GTW_SESSION_START Request parameters

FRMCS_GTW_SESSION_START Response parameters

FIELD	TYPE	VALUE
jsonrpc	String	2.0
id	String	same value than in the FRMCS_GTW_SESSION_START request
result	Object	See next table.

Table 14 TC_003 FRMCS_GTW_SESSION_START Response JsonRPC parameters

FIELD	TYPE	VALUE
session_uuid	String	A new local ID unique chosen by itself if the request is succeeded. Must be build following RFC4122
ip_dest	String	IP address to be used by the application as destination address for the user plane data

Table 15 TC_003 FRMCS_GTW_SESSION_START Response parameters

FRMCS_APP_SESSION_STATUS_CHANGED notification parameters

FIELD	TYPE	VALUE
jsonrpc	String	2.0
result	Object	See next table.

Table 16 TC_003 FRMCS_APP_SESSION_STATUS_CHANGED notification JsonRPC parameters

FIELD	TYPE	VALUE
session_uuid	String	The same session_uuid value obtained in the FRMCS_GTW_SESSION_START Response.
session_status	String	trying (in the first notification message) working (in the second notification message)
details	String	RAS

Table 17 TC_003 FRMCS_APP_SESSION_STATUS_CHANGED notification parameters

3.2.3.4 Success criteria

- All partial results are as expected.
- No error messages

3.2.4 TC_004: Session end request of an application with FRMCS onboard Gateway

3.2.4.1 Purpose

The purpose of this test is to validate the ability to launch a session finalization request in the FRMCS network.

3.2.4.2 Description of initial state/configuration

The initial state covers the following steps:

- Application onboard equipment are installed and configured
- FRMCS Onboard Gateway is connected and configured to application equipment
- Application equipment are connected and power on in nominal state
- Application is simulated to be registered in the FRMCS network via FRCMS Onboard Gateway.
- A session is simulated to be established in FRMCS network via FRCMS Onboard Gateway.
- Websocket already established and registration already performed

3.2.4.3 Test procedure: Successful ending of an application session

STEP	ACTION	EXPECTED RESULT(S)
1	FRMCS_GTW_SESSION_END Request JSON message is sent to FRMCS Gateway with the specified values. (See next tables).	- FRMCS_GTW_SESSION_END Response JSON message is received with the specified values. (See next tables). - The session is removed from the FRMCS network with its uuid.

Table 18 TC_004 Test procedure

FRMCS_GTW_SESSION_END Request parameters

FIELD	TYPE	VALUE
jsonrpc	String	2.0
method	String	session_end
id	String	Implementation dependant
params	Object	See next table.

Table 19 TC_004 FRMCS_GTW_SESSION_END Request JsonRPC parameters

FIELD	TYPE	VALUE
app_uuid	String	Pre-established app uuid.
session_uuid	String	Pre-established session uuid.

Table 20 TC_004 FRMCS_GTW_SESSION_END Request parameters

FRMCS_GTW_SESSION_END Response parameters

FIELD	TYPE	VALUE
jsonrpc	String	2.0
id	String	same value than in the FRMCS_GTW_SESSION_END request
result	Object	Empty

Table 21 TC_004 FRMCS_GTW_SESSION_END JsonRPC parameters

FIELD	TYPE	VALUE
N/A	N/A	N/A

Table 22 TC_004 FRMCS_GTW_SESSION_END Response parameters

3.2.4.4 Success criteria

- All partial results are as expected
- No error messages.

3.2.5 TC_005: Session status request of an application with FRMCS onboard Gateway

3.2.5.1 Purpose

The purpose of this test is to validate the ability to launch a session status request to the FRMCS network.

3.2.5.2 Description of initial state/configuration

The initial state covers the following steps:

- Application onboard equipment are installed and configured
- FRMCS Onboard Gateway is connected and configured to application equipment
- Application equipment are connected and power on in nominal state
- The application is simulated to be registered in the FRMCS network via FRCMS Onboard Gateway.
- A session is simulated to be established in FRMCS network via FRCMS Onboard Gateway.
- Websocket already established and registration already performed

3.2.5.3 Test procedure: Successful status request of an application session

STEP	ACTION	EXPECTED RESULT(S)
1	FRMCS_GTW_SESSION_STATUS Request JSON message is sent to FRMCS Gateway with the specified values. (See next tables).	- FRMCS_GTW_SESSION_STATUS Response JSON message is received with the specified values. (See next tables). - A valid session status is returned.

Table 23 TC_005 Test procedure

FRMCS_GTW_SESSION_STATUS Request Json Structure

FIELD	TYPE	VALUE
jsonrpc	String	2.0
method	String	session_status
id	String	Implementation dependant
params	Object	See next table.

Table 24 TC_005 FRMCS_GTW_SESSION_STATUS Request JsonRPC parameters

FIELD	TYPE	VALUE
app_uuid	String	A pre-established simulated application uuid.
session_uuid	Object	A pre-established simulated session uuid.

Table 25 TC_005 FRMCS_GTW_SESSION_STATUS Request parameters

FRMCS_GTW_SESSION_STATUS Response parameters

FIELD	TYPE	VALUE
jsonrpc	String	2.0
id	String	same value than in the FRMCS_GTW_SESSION_STATUS request
result	Object	See next table.

Table 26 TC_005 FRMCS_GTW_SESSION_STATUS Response JsonRPC parameters

FIELD	TYPE	VALUE
status	String	working
details	String	RAS

Table 27 TC_005 FRMCS_GTW_SESSION_STATUS Response parameters

3.2.5.4 Success criteria

- All partial results are as expected.
- No error messages

3.2.6 TC_006: Service request with FRMCS onboard Gateway

3.2.6.1 Purpose

The purpose of this test is to validate the ability to request a service to the FRMCS network.

3.2.6.2 Description of initial state/configuration

The initial state covers the following steps:

- Application onboard equipment are installed and configured
- FRMCS Onboard Gateway is connected and configured to application equipment
- Application equipment are connected and power on in nominal state
- The application is simulated to be registered in the FRMCS network via FRCMS Onboard Gateway.
- Websocket already established and registration already performed

3.2.6.3 Test procedure: Successful service request

STEP	ACTION	EXPECTED RESULT(S)
1	FRMCS_GTW_SERVICE_REQUEST Request JSON message is sent to FRMCS Gateway with the specified values. (See next tables).	- FRMCS_GTW_SERVICE_REQUEST response JSON message is received with the specified values. (See next tables). - The service request response contains the specified value (See next tables).

Table 28 TC_006 Test procedure

FRMCS_GTW_SERVICE_REQUEST Request Json Structure

FIELD	TYPE	VALUE
jsonrpc	String	2.0
method	String	service_request
Id	String	Implementation dependant
params	Object	See next table.

Table 29 TC_006 FRMCS_GTW_SERVICE_REQUEST Request JsonRPC parameters

FIELD	TYPE	VALUE
app_uuid	String	A pre-established simulated application uuid.
request_type	String	connection_status Note: Only connection_status will be tested. Request for gnss positioning not implemented in 5GRail.

Table 30 TC_006 FRMCS_GTW_SERVICE_REQUEST Request parameters

FRMCS_GTW_SERVICE_REQUEST Response parameters

FIELD	TYPE	VALUE
jsonrpc	String	2.0
id	String	same value than in the FRMCS_GTW_SERVICE_REQUEST request
result	Object	See next table.

Table 31 TC_006 FRMCS_GTW_SERVICE_REQUEST Response JsonRPC parameters

FIELD	TYPE	VALUE
connection_status	String	connected

Table 32 TC_006 FRMCS_GTW_SERVICE_REQUEST Response parameters

3.2.6.4 Success criteria

- All partial results are as expected.
- No error messages

3.2.7 TC_007: Incoming session request from the FRMCS onboard Gateway

3.2.7.1 Purpose

The purpose of this test is to validate the ability to receive the incoming session requests coming from the FRMCS gateway.

Note: This test is only supported by the “phase 2” FRMCS GW simulator (see chapter 2.3 for more information).

3.2.7.2 Description of initial state/configuration

The initial state covers the following steps:

- Application onboard equipment are installed and configured
- FRMCS Onboard Gateway is connected and configured to application equipment
- Application equipment are connected and power on in nominal state
- The application is simulated to be registered in the FRMCS network via FRCMS Onboard Gateway.
    - Note: The application shall be registered with the “not_auto” value in the registration request parameter.
- Websocket already established and registration already performed

3.2.7.3 Test procedure: Successful service request

STEP	ACTION	EXPECTED RESULT(S)
1	The registered Trackside application sends a FRMCS_GTW_SESSION_START request to the Simulator, on the “TS interface”. The target of this session request is the registered name of the on-board application.	The TS application receives a FRMCS_GTW_SESSION_START answers, with a session_uuid and an IP address to be used for user plane. The on-board application receives a FRMCS_APP_INCOMING_SESSION_REQ with a session_uuid (which can be different than the one received by the TS part), the originator of the call, and an IP address to be used for user plane.
2	The on-board application answers to the FRMCS_APP_INCOMING_SESSION_REQ to accept the incoming session.	The OB application receives an update of the session status (working”) within a FRMCS_APP_SESSION_STATUS_CHANGED notification. The TS application receives an update of the session status (“working”) within a FRMCS_APP_SESSION_STATUS_CHANGED notification.

Table 33 TC_006 Test procedure

FRMCS_APP_INCOMING_SESSION_REQ Request Json Structure

FIELD	TYPE	VALUE
jsonrpc	String	2.0
method	String	Incoming_session
Id	String	Implementation dependant
params	Object	See next table.

Table 34 TC_007 FRMCS_APP_INCOMING_SESSION_REQ Request JsonRPC parameters

FIELD	TYPE	VALUE
session_uuid	String	A pre-established simulated application uuid.
source	String	A new local ID unique chosen by itself if the request is succeeded. Must be build following RFC4122
ip_src	String	IP address of the incomming session application

Table 35 TC_007 FRMCS_APP_INCOMING_SESSION_REQ Request parameters

FRMCS_APP_INCOMING_SESSION_REQ Response parameters

FIELD	TYPE	VALUE
jsonrpc	String	2.0
id	String	same value than in the FRMCS_APP_INCOMING_SESSION_REQ request
result	Object	See next table.

Table 36 TC_007 FRMCS_APP_INCOMING_SESSION_REQ Response JsonRPC parameters

FIELD	TYPE	VALUE
Return	String	“OK”

Table 37 TC_007 FRMCS_APP_INCOMING_SESSION_REQ Response parameters

3.2.7.4 Success criteria

- All partial results are as expected.
- No error messages

3.2.8 TC_008: Incoming session notification from the FRMCS onboard Gateway

3.2.8.1 Purpose

The purpose of this test is to validate the ability to receive the incoming session notification coming from the FRMCS gateway.

Note: This test is only supported by the “phase 2” FRMCS GW simulator (see chapter 2.3 for more information), or latest Alstom simulator referenced “Simu_Ph1.v2.1” (see chapter **2.2.1.2**).

3.2.8.2 Description of initial state/configuration

The initial state covers the following steps:

- Application onboard equipment are installed and configured
- FRMCS Onboard Gateway is connected and configured to application equipment
- Application equipment are connected and power on in nominal state
- The application is simulated to be registered in the FRMCS network via FRCMS Onboard Gateway.
    - Note: The application shall be registered with the “auto_accept” value in the registration request parameter.
- Websocket already established and registration already performed in both sides (connection originator and receptor)

3.2.8.3 Test procedure: Successful service request

STEP	ACTION	EXPECTED RESULT(S)
1	The registered Trackside application sends a FRMCS_GTW_SESSION_START request to the Simulator, on the “TS interface”. The target of this session request is the registered name of the on-board application.	The TS application receives a FRMCS_GTW_SESSION_START answers, with a session_uuid and an IP address to be used for user plane. The on-board application receives a FRMCS_APP_INCOMING_SESSION_NOTIF with a session_uuid (which can be different than the one received by the TS part), the originator of the call, and an IP address to be used for user plane. Then, as the called application is in auto_accept mode, the session goes into the state “working” almost immediately. Then, both OB and TS application receives a notification for session status (with “working” state).

Table 38 TC_006 Test procedure

FRMCS_APP_INCOMING_SESSION_NOTIF Json Structure

FIELD	TYPE	VALUE
jsonrpc	String	2.0
method	String	incoming_session_notif
params	Object	See next table.

Table 39 TC_008 FRMCS_APP_INCOMING_SESSION_NOTIF JsonRPC parameters

FIELD	TYPE	VALUE
session_uuid	String	A pre-established simulated application uuid.
source	String	A new local ID unique chosen by itself if the request is succeeded. Must be build following RFC4122
ip_src	String	IP address of the incomming session application

Table 40 TC_008 FRMCS_APP_INCOMING_SESSION_NOTIF parameters

3.2.8.4 Success criteria

- All partial results are as expected.
- No error messages

3.3 Tight Coupled Applications (Siemens)

3.3.1 TC_001: Registration of an application with FRMCS onboard Gateway

3.3.1.1 Purpose

The purpose of this test is to validate the ability to register an application in the FRMCS network.

3.3.1.2 Description of initial state/configuration

The initial state covers the following steps:

- Application onboard equipment installed and configured
- FRMCS Onboard Gateway is connected and configured to application equipment
- Application equipment is connected and power on in nominal state

3.3.1.3 Test procedure: Successful registration of an application

STEP	ACTION	EXPECTED RESULT(S)
1	Open a WebSocket connection with the gateway.	The connection is established successfully.
2	FRMCS_GTW_REGISTER Request JSON message is sent to FRMCS Gateway with the specified values. (See next tables).	- FRMCS_GTW_REGISTER Response JSON message is received with the specified values. (See next tables). - Application with its uuid is registered in the FRMCS network.

Table 41 TC_001 Test procedure

FRMCS_GTW_REGISTER Request Json Structure

FIELD	TYPE	VALUE
Jsonrpc	String	2.0
Method	String	register
Id	String	Implementation dependant
Params	Object	See next table.

Table 42 TC_001 FRMCS_GTW_REGISTER Request JsonRPC parameters

FIELD	TYPE	VALUE
application_type	Int	3 (Voice)
originator_id	String	ID of the host (XXX.XXX.XX) or FQDN format
Mode	String	Tight
incoming_auto	String	not_auto (Not used by tight coupled applications)

Table 43 TC_001 FRMCS_GTW_REGISTER Request parameters

FRMCS_GTW_REGISTER Response parameters

FIELD	TYPE	VALUE
Jsonrpc	String	2.0
id	String	same value than in the FRMCS_GTW_REGISTER request
result	Object	See next table.

Table 44 TC_001 FRMCS_GTW_REGISTER Response JsonRPC parameters

FIELD	TYPE	VALUE
app_uuid	String	A new local ID unique chosen by itself if the request is succeeded. Must be build following RFC4122

Table 45 TC_001 FRMCS_GTW_REGISTER Response parameters

3.3.1.4 Success criteria

- All partial results are as expected.
- No error messages

3.3.2 TC_002: Deregistration of an application with FRMCS onboard Gateway

3.3.2.1 Purpose

The purpose of this test is to validate the ability to deregister an application in the FRMCS network.

3.3.2.2 Description of initial state/configuration

The initial state covers the following steps:

- Application onboard equipment are installed and configured
- FRMCS Onboard Gateway is connected and configured to application equipment
- Application equipment are connected and power on in nominal state
- Application is simulated to be registered in the FRMCS network via FRCMS Onboard Gateway
- Websocket already established and registration already performed

3.3.2.3 Test procedure: Successful deregistration of an application

STEP	ACTION	EXPECTED RESULT(S)
1	FRMCS_GTW_DEREGISTER Request JSON message is sent to FRMCS Gateway with the specified values. (See next tables).	- FRMCS_GTW_DEREGISTER Response JSON message is received with the specified values. (See next tables). - Application with its uuid is deregistered from the FRMCS network

Table 46 TC_002 Test procedure

FRMCS_GTW_DEREGISTER Request parameters

FIELD	TYPE	VALUE
jsonrpc	String	2.0
method	String	deregister
id	String	Implementation dependant
params	Object	See next table.

Table 47 TC_002 FRMCS_GTW_DEREGISTER Request JsonRPC parameters

FIELD	TYPE	VALUE
app_uuid	String	Uuid of the application

Table 48 TC_002 FRMCS_GTW_DEREGISTER Request parameters

FRMCS_GTW_DEREGISTER Response parameters

FIELD	TYPE	VALUE
N/A	N/A	N/A

Table 49 TC_002 FRMCS_GTW_DEREGISTER Response parameters

3.3.2.4 Success criteria

- All partial results are as expected
- No error messages.

3.3.3 TC_003: Service request with FRMCS onboard Gateway

3.3.3.1 Purpose

The purpose of this test is to validate the ability to request a service to the FRMCS network.

3.3.3.2 Description of initial state/configuration

The initial state covers the following steps:

- Application onboard equipment are installed and configured
- FRMCS Onboard Gateway is connected and configured to application equipment
- Application equipment are connected and power on in nominal state
- The application is simulated to be registered in the FRMCS network via FRCMS Onboard Gateway
- Websocket already established and registration already performed

3.3.3.3 Test procedure: Successful service request

STEP	ACTION	EXPECTED RESULT(S)
1	FRMCS_GTW_SERVICE_REQUEST Request JSON message is sent to FRMCS Gateway with the specified values. (See next tables).	- FRMCS_GTW_SERVICE_REQUEST response JSON message is received with the specified values. (See next tables). - The service request response contains the specified value (See next tables).
2	The application registers the operation result.	

Table 50 TC_003 Test procedure

FRMCS_GTW_SERVICE_REQUEST Request Json Structure

FIELD	TYPE	VALUE
jsonrpc	String	2.0
method	String	service_request
Id	String	Implementation dependant
params	Object	See next table.

Table 51 TC_003 FRMCS_GTW_SERVICE_REQUEST Request JsonRPC parameters

FIELD	TYPE	VALUE
app_uuid	String	A pre-established simulated application uuid.
request_type	String	connection_status

Table 52 TC_003 FRMCS_GTW_SERVICE_REQUEST Request parameters

FRMCS_GTW_SERVICE_REQUEST Response parameters

FIELD	TYPE	VALUE
jsonrpc	String	2.0
id	String	same value than in the FRMCS_GTW_SERVICE_REQUEST request
result	Object	See next table.

Table 53 TC_003 FRMCS_GTW_SERVICE_REQUEST Response JsonRPC parameters

FIELD	TYPE	VALUE
connection_status	String	connected

Table 54 TC_003 FRMCS_GTW_SERVICE_REQUEST Response parameters

3.3.3.4 Success criteria

- All partial results are as expected.
- No error messages

4 Test Results

4.1 Introduction

This chapter contains the results of the pre-integration test execution by each application supplier. The chapter contains a summary table where each application provider indicates if the test case passed, failed or is not applicable (some applications might not use some OBapp features). Moreover, there is another subchapter named "Justifications", where each application supplier indicates the specificities of the test exception, possible clarifications or deviations that were detected during the test execution.

Due to remote access restrictions, phase 2 simulator has only been accessible locally for WP4 lab partners (Alstom and Thales). Therefore, the test cases TC07 and TC08 are only tested in WP4 applications.

4.2 Result table

Loose coupled applications:

Applications	TC_001	TC_002	TC_003	TC_004	TC_005	TC_006	TC_007	TC_008
Alstom – ATO (ETCS/ATP)	OK* (Phase 1 and phase 2 simulator)	OK (Phase 1 and phase 2 simulator)	OK (Phase 1 and phase 2 simulator)	OK (Phase 1 and phase 2 simulator)	OK (Phase 1 and phase 2 simulator)	OK (Phase 1 and phase 2 simulator)	OK* (Phase 2 simulator)	OK (Ph 1.v2.1 simulator)
CAF – ETCS/ATP&TCMS (See Note 2)	OK* (Phase 1 simulator)	OK (Phase 1 simulator)	OK (Phase 1 simulator)	OK (Phase 1 simulator)	OK (Phase 1 simulator)	OK (Phase 1 simulator)	N/A	N/A
Teleste – CCTV Offload	OK (Phase 1 simulator)	OK (Phase 1 simulator)	OK (Phase 1 simulator)	OK (Phase 1 simulator)	OK (Phase 1 simulator)	OK (Phase 1 simulator)	N/A	N/A
Teleste – Video	OK (Phase 1 simulator)	OK (Phase 1 simulator)	OK (Phase 1 simulator)	OK (Phase 1 simulator)	OK (Phase 1 simulator)	OK (Phase 1 simulator)	N/A	N/A
Thales – PIS	OK (Phase 1 & phase 2 simulator)	OK (Phase 1 & phase 2 simulator)	OK (Phase 1 & phase 2 simulator)	OK (Phase 1 & phase 2 simulator)	OK (Phase 1 simulator) NOK[4] (Phase 2 simulator) or)	OK (Phase 1 simulator) NOK[4] (Phase 2 simulator)	Not executed[3]	Not executed[3]

Note 1: The results that contain an asterisk "*" include a justification in the subchapter 4.3. The columns including N/A mean that test cases are not applicable, because those tests can only be tested with phase 2 simulator, and only the phase 1 simulator was available for WP3 applications.

Note 2: CAF ETCS and TCMS simulators will share the same OBApp/TSApp implementations, only application part is changed. Therefore, these tests are executed directly using the common library.

Note 3: TC_007 and TC_008 was not defined at the stage of Thales and Phase 2 simulator testing sessions. To be tested and executed on the next testing session

Note 4: See more details on Thales justifications section (4.3.4)

Tight coupled applications:

Test Case	TC_001	TC_002	TC_003
Siemens - Voice	OK (phase 1 simulator)	OK (phase 1 simulator)	OK (phase 1 simulator)

4.3 Justifications

The following chapter includes the justification of the test case execution, including the log/trace files that demonstrate it.

4.3.1 Alstom

Test Case	TC_001
Remarks	Only "auto_accept" mode is used with Phase 1 simulator (Ph1.V1.2 and Ph1.V2.1) Only "not_auto" mode is used with Phase 2 simulator. The field "return" should be deleted (not in the spec)
Attachments (log/trace file)	See below

Test Case	TC_007
Remarks	Name of the request should be “incoming_session_req” instead of “incoming_session”. The “src” parameter should contain the ID of the caller. The IP address to be used in user plane should be given in “ip_src”.
Attachments (log/trace file)	See below

For all test cases:

- The tests with Phase 1 simulator are saved in the file named “5GRAIL_WP2_Alstom_ATO_Ph1_simu.zip”.
- The test with Phase 2 simulator are saved in the filed named: “5GRAIL_WP4_Alstom_testATO_2022-03-10.zip”.

4.3.2 CAF

Test Case	TC_001
Specific test configuration	Application: CAFS_FRMCS_simulator v1 Simulator: Alstom Simu_Ph1.v1.2
Specific test conditions	Executed in three application configurations: Incomming_auto: “auto_reject” Incomming_auto: “auto_accept” Incomming_auto: “not_auto”
Remarks	Registration type “auto_accept” and “auto_reject” worked properly. The type “not_auto” is not supported by the gateway simulator and returns an error (“invalid incoming_auto”). It is understood that it is not an error, but a limitation of the simulator.
Tested by	CAFS
Attachments (log/trace file)	TC001_auto_reject.pcap TC001_auto_accept.pcap TC001_not_auto.pcap
Test result	Passed

Test Case	TC_002
Specific test configuration	Application: CAFS_FRMCS_simulator v1 Simulator: Alstom Simu_Ph1.v1.2
Specific test conditions	None
Remarks	No issues identified.
Tested by	CAFS
Attachments **(log/trace file)**	TC002.pcap
Test result	Passed

Test Case	TC_003
Specific test configuration	Application: CAFS_FRMCS_simulator v1 Simulator: Alstom Simu_Ph1.v1.2
Specific test conditions	None
Remarks	No issues identified.
Tested by	CAFS
Attachments **(log/trace file)**	TC003.pcap
Test result	Passed

Test Case	TC_004
Specific test configuration	Application: CAFS_FRMCS_simulator v1 Simulator: Alstom Simu_Ph1.v1.2
Specific test conditions	None
Remarks	No issues identified.
Tested by	CAFS
Attachments **(log/trace file)**	TC004.pcap
Test result	Passed

Test Case	TC_005
Specific test configuration	Application: CAFS_FRMCS_simulator v1 Simulator: Alstom Simu_Ph1.v1.2
Specific test conditions	None
Remarks	No issues identified.
Tested by	CAFS
Attachments **(log/trace file)**	TC005.pcap
Test result	Passed

Test Case	TC_006
Specific test configuration	Application: CAFS_FRMCS_simulator v1 Simulator: Alstom Simu_Ph1.v1.2
Specific test conditions	Executed with the following configuration: request_type = connection_status
Remarks	No issues identified.
Tested by	CAFS
Attachments **(log/trace file)**	TC006.pcap
Test result	Passed

4.3.3 Teleste

Test Case	TC_001
Specific test configuration	Application: Teleste_FRMCS_client_sample v1 CCTV offload: • "application_type":6," • “originator_id":"nvr1.001.ob.cctv" Simulator: Alstom Simu_Ph1.v1.2
Specific test conditions	Executed in application configuration: Incomming_auto: “auto_accept”
Remarks	Registration type “auto_accept” worked properly.
Tested by	TELESTE
Attachments (log/trace file)	Log-Simu_Ph1.v1.2_CCTV-Offload.txt CCTV-Offload.pcapng
Test result	Passed

Test Case	TC_001
Specific test configuration	Application: Teleste_FRMCS_client_sample v1 CCTV live: • "application_type":7," • “originator_id":"wcg1.001.ts.cctv" Simulator: Alstom Simu_Ph1.v1.2
Specific test conditions	Executed in application configuration: Incomming_auto: “auto_accept”
Remarks	Registration type “auto_accept” worked properly.
Tested by	TELESTE
Attachments (log/trace file)	Log-Simu_Ph1.v1.2_CCTV-Live.txt CCTV-Live.pcapng
Test result	Passed

Test Case	TC_002
Specific test configuration	Application: Teleste_FRMCS_client_sample v1 CCTV offload: • "application_type":6," • “originator_id":"nvr1.001.ob.cctv" Simulator: Alstom Simu_Ph1.v1.2
Specific test conditions	None
Remarks	No issues identified.
Tested by	TELESTE
Attachments (log/trace file)	Log-Simu_Ph1.v1.2_CCTV-Offload.txt CCTV-Offload.pcapng
Test result	Passed

Test Case	TC_002
Specific test configuration	Application: Teleste_FRMCS_client_sample v1 CCTV live: • "application_type":7," • “originator_id":"wcg1.001.ts.cctv" Simulator: Alstom Simu_Ph1.v1.2
Specific test conditions	None
Remarks	No issues identified.
Tested by	TELESTE
Attachments (log/trace file)	Log-Simu_Ph1.v1.2_CCTV-Live.txt CCTV-Live.pcapng
Test result	Passed

Test Case	TC_003
Specific test configuration	Application: Teleste_FRMCS_client_sample v1 CCTV offload: • "application_type":6," • “originator_id":"nvr1.001.ob.cctv" Simulator: Alstom Simu_Ph1.v1.2
Specific test conditions	None
Remarks	No issues identified.
Tested by	TELESTE
Attachments (log/trace file)	Log-Simu_Ph1.v1.2_CCTV-Offload.txt CCTV-Offload.pcapng
Test result	Passed

Test Case	TC_003
Specific test configuration	Application: Teleste_FRMCS_client_sample v1 CCTV live: • "application_type":7," • “originator_id":"wcg1.001.ts.cctv" Simulator: Alstom Simu_Ph1.v1.2
Specific test conditions	None
Remarks	No issues identified.
Tested by	TELESTE
Attachments (log/trace file)	Log-Simu_Ph1.v1.2_CCTV-Live.txt CCTV-Live.pcapng
Test result	Passed

Test Case	TC_004
Specific test configuration	Application: Teleste_FRMCS_client_sample v1 CCTV offload: • "application_type":6," • “originator_id":"nvr1.001.ob.cctv" Simulator: Alstom Simu_Ph1.v1.2
Specific test conditions	None
Remarks	No issues identified.
Tested by	TELESTE
Attachments (log/trace file)	Log-Simu_Ph1.v1.2_CCTV-Offload.txt CCTV-Offload.pcapng
Test result	Passed

Test Case	TC_004
Specific test configuration	Application: Teleste_FRMCS_client_sample v1 CCTV live: • "application_type":7," • “originator_id":"wcg1.001.ts.cctv" Simulator: Alstom Simu_Ph1.v1.2
Specific test conditions	None
Remarks	No issues identified.
Tested by	TELESTE
Attachments (log/trace file)	Log-Simu_Ph1.v1.2_CCTV-Live.txt CCTV-Live.pcapng
Test result	Passed

Test Case	TC_005
Specific test configuration	Application: Teleste_FRMCS_client_sample v1 CCTV offload: • "application_type":6," • “originator_id":"nvr1.001.ob.cctv" Simulator: Alstom Simu_Ph1.v1.2
Specific test conditions	None
Remarks	No issues identified.
Tested by	TELESTE
Attachments (log/trace file)	Log-Simu_Ph1.v1.2_CCTV-Offload.txt CCTV-Offload.pcapng
Test result	Passed

Test Case	TC_005
Specific test configuration	Application: Teleste_FRMCS_client_sample v1 CCTV live: • "application_type":7," • “originator_id":"wcg1.001.ts.cctv" Simulator: Alstom Simu_Ph1.v1.2
Specific test conditions	None
Remarks	No issues identified.
Tested by	TELESTE
Attachments (log/trace file)	Log-Simu_Ph1.v1.2_CCTV-Live.txt CCTV-Live.pcapng
Test result	Passed

Test Case	TC_006
Specific test configuration	Application: Teleste_FRMCS_client_sample v1 CCTV offload: • "application_type":6," • “originator_id":"nvr1.001.ob.cctv" Simulator: Alstom Simu_Ph1.v1.2
Specific test conditions	Executed with the following configuration: request_type = connection_status
Remarks	No issues identified.
Tested by	TELESTE
Attachments (log/trace file)	Log-Simu_Ph1.v1.2_CCTV-Offload.txt CCTV-Offload.pcapng
Test result	Passed

Test Case	TC_006
Specific test configuration	Application: Teleste_FRMCS_client_sample v1 CCTV live: • "application_type":7," • “originator_id":"wcg1.001.ts.cctv" Simulator: Alstom Simu_Ph1.v1.2
Specific test conditions	Executed with the following configuration: request_type = connection_status
Remarks	No issues identified.
Tested by	TELESTE
Attachments (log/trace file)	Log-Simu_Ph1.v1.2_CCTV-Live.txt CCTV-Live.pcapng
Test result	Passed

4.3.4 Thales

These test cases have been executed with the Alstom' simulator which simulates OBapp/TSapp "server" normally embbeded in the FRMCS Gateways and Kontron's FRMCS robot.

The program frmcsAgent is part of PIS application and is responsible of the OBapp/TSapp exhanges as specified in D2.1 v3.

At this stage, Kontron's FRMCS robot is not compliant with D2.1 REV 3.1 where the FRMCS primitive FRMCS_APP_INCOMING_SESSION_REQ has been updated with the addition of the parameter "ip_src". Therefore, the FRMCS registration process is only performed in "auto_accept" mode.

Test Case	TC_001
Specific test configuration	Alstom'simulator v1
Specific test conditions	None
Remarks	None
Tested by	Thales
Attachments **(log/trace file)**	TSapp_OBapp_loose_registration_20220126.log pre-integration_FRMCS_loose_tests_TS.pcapng
Test result	Passed

Test Case	TC_001
Specific test configuration	Kontron's FRMCS robot
Specific test conditions	None
Remarks	None
Tested by	Kontron Thales
Attachments **(log/trace file)**	5GRail_WP2_Thales_Test_Robot_Kontron_20220222_TS_TC_001.pcapng 5GRail_WP2_Thales_Test_Robot_Kontron_20220222_TS_TC_001.log 5GRail_WP2_Thales_Test_Robot_Kontron_20220222_OB_TC_001.pcapng 5GRail_WP2_Thales_Test_Robot_Kontron_20220222_OB_TC_001.log
Test result	Passed

Test Case	TC_002
Specific test configuration	Alstom'simulator v1
Specific test conditions	None
Remarks	None
Tested by	Thales
Attachments **(log/trace file)**	TSapp_OBapp_loose_deregistration_20220126.log pre-integration_FRMCS_loose_tests_TS.pcapng
Test result	Passed

Test Case	TC_002
Specific test configuration	Kontron's FRMCS robot
Specific test conditions	None
Remarks	None
Tested by	Kontron Thales
Attachments **(log/trace file)**	5GRail_WP2_Thales_Test_Robot_Kontron_20220222_TS_TC_002.pcapng 5GRail_WP2_Thales_Test_Robot_Kontron_20220222_TS_TC_002.log 5GRail_WP2_Thales_Test_Robot_Kontron_20220222_OB_TC_002.log
Test result	Passed

Test Case	TC_003
Specific test configuration	Alstom'simulator v1
Specific test conditions	None
Remarks	None
Tested by	Thales
Attachments **(log/trace file)**	TSapp_OBapp_loose_start_session_20220126.log pre-integration_FRMCS_loose_tests_TS.pcapng
Test result	Passed

Test Case	TC_003
Specific test configuration	Kontron's FRMCS robot
Specific test conditions	None
Remarks	None
Tested by	Kontron Thales
Attachments **(log/trace file)**	5GRail_WP2_Kontron_Robot_Thales_PIS_D2_2_20220228_TS.pcapng 5GRail_WP2_Kontron_Robot_Thales_PIS_D2_2_20220228_TS.txt
Test result	Passed

Test Case	TC_004
Specific test configuration	Alstom'simulator v1
Specific test conditions	None
Remarks	None
Tested by	Thales
Attachments **(log/trace file)**	TSapp_OBapp_loose_end_session_20220127.log pre-integration_FRMCS_loose_tests_TS_20220127.pca png
Test result	Passed

Test Case	TC_004
Specific test configuration	Kontron's FRMCS robot
Specific test conditions	None
Remarks	None
Tested by	Kontron Thales
Attachments **(log/trace file)**	TSapp_OBapp_loose_end_session_20220127.log pre-integration_FRMCS_loose_tests_TS_20220127.pca png
Test result	Passed

Test Case	TC_005
Specific test configuration	Alstom'simulator v1
Specific test conditions	None
Remarks	None
Tested by	Thales
Attachments **(log/trace file)**	TSapp_OBapp_loose_session_status_20220126.log pre-integration_FRMCS_loose_tests_TS.pcapng
Test result	Passed

Test Case	TC_005
Specific test configuration	Kontron's FRMCS robot
Specific test conditions	None
Remarks	None
Tested by	Kontron Thales
Attachments **(log/trace file)**	5GRail_WP2_Kontron_Robot_Thales_PIS_D2_2_20220228_TS.pcapng 5GRail_WP2_Kontron_Robot_Thales_PIS_D2_2_20220228_TS.txt
Test result	Failed. Session opened by the PIS trackside application is in "working" state but when the PIS trackside application requests a "session_status" the FRMCS trackside robot answers that the session is in "trying" state instead of "working".

Test Case	TC_006
Specific test configuration	Alstom'simulator v1
Specific test conditions	None
Remarks	None
Tested by	Thales
Attachments **(log/trace file)**	TSapp_OBapp_loose_service_request_20220127.log pre-integration_FRMCS_loose_tests_TS_20220127.pcapng
Test result	Passed

Test Case	TC_006
Specific test configuration	Kontron's FRMCS robot
Specific test conditions	None
Remarks	At this stage, this test cannot be executed because Kontron's robot does not implement the method "service_request".
Tested by	Not Applicable
Attachments **(log/trace file)**	None
Test result	Not executed

4.3.5 Siemens

Test Case	TC_001
Remarks	Tested on radio using phase 1 simulator.
Tested by	Siemens
Attachments (log/trace file)	None
Test result	Passed

Test Case	TC_002
Remarks	Tested on radio using phase 1 simulator.
Tested by	Siemens
Attachments (log/trace file)	None
Test result	Passed

Test Case	TC_003
Remarks	Tested on radio using phase 1 simulator.
Tested by	Siemens
Attachments (log/trace file)	None
Test result	Passed

5 Additional Test Cases (Optional)

5.1 Introduction

As it is stated in the introduction of the document, the main objective of the pre-integration phase is to de-risk the lab testing phase, by testing the integration between the applications and the FRMCS Gateways (on-board and trackside). However, the pre-integration framework will allow the applications to test additional test cases that will also de-risk the lab implementation. Therefore, the objective of this chapter is to allow each application provider to describe the test cases that will be tested in this pre-integration phase.

5.2 ETCS/ATP application (Alstom)

This chapter aims at selecting additional pre-integration tests performed in Alstom labs, among the tests described in chapter 8.2 of D1.1 document.

5.2.1 Phasing

At this stage, two phases are considered for Alstom-ETCS:

- Phase 1: application in Flat-IP only
    - Pre-integration performed in Alstom labs beginning of August 2021.
    - Integration in Kontron labs (WP4) in November 2021
- Phase 2: Loose-coupling approach, OBAPP implemented for OB part
    - Pre-integration in Alstom labs beginning of March 2022
    - Integration in Kontron labs (WP4) in May 2022

Hence, only phase 1 is considered for the first delivery to WP4 in November 2021.

5.2.2 Additional pre-integration tests - Phase 1

TC_001 – Nominal communication between ETCS OB application and RBC:

The corresponding tests are described in chapter 8.2.1.1 of D1.1. Some of them are also performed during pre-integration stage in Alstom labs, as explained below:

DESCRIPTION OF THE TESTS TO BE DONE IN WP4	CORRESPONDING D1.1 CHAPTER	STATUS OF THE CORRESPONDING PRE-INTEGRATION TEST IN ALSTOM LABS
Establish a communication between ETCS On-board and RBC. Check communication are correctly set Check the correct data transfer Perform an end of communication	8.2.1.1.3	It is fully performed during pre-integration tests.
Perform a RBC handover (using a balise scenario) Perform a BTS handover	8.2.1.1.4	Only RBC handover is performed during pre-integration tests.
Endurance tests: Perform multiple RBCs handover (using a balise scenario) during several hours (approximately 6 or 8h).	8.2.1.1.5	Not performed during pre-integration tests

Table 55 TC_001 – Nominal communication between OB-ETCS and RBC

TC_002 – Bearer flexibility:

The corresponding tests are described in chapter 8.2.1.2 of D1.1. The purpose is to check that the communication is not impacted by a switch of radio bearer.

They are also partly performed at pre-integration stage in Alstom labs, as explained below:

DESCRIPTION OF THE TESTS TO BE DONE IN WP4	CORRESPONDING D1.1 CHAPTER	STATUS OF THE CORRESPONDING PRE-INTEGRATION TEST IN ALSTOM LABS
After a correct communication establishment between OB and TS, trigger a mobile switch on a second radio bearer. Check that the data transfer is still on-going and the communication is not impacted by the switch.	8.2.1.2.3	It is performed during pre-integration tests. The nominal communication is on 5G channel. We use attenuator to simulate a loss of 5G network, and switch to a 4G link then Wi-Fi link.

Table 56 TC_002 – Bearer flexibility Test cases

5.2.3 Additional pre-integration tests – Phase 2

For the first WP4 HW and SW delivery in November 2021 in Kontron labs, the pre-integration tests related to phase 2 will not be performed.

A revision could be done for the second delivery to WP4 to include Phase 2 tests.

5.3 ATO application (Alstom)

This chapter aims at selecting additional pre-integration tests performed in Alstom labs, among the tests described in chapter 8.3 of D1.1 document.

Basically, it aims at running some test scenario requiring a communication between ATO on-board and trackside equipment (ATO-OB and ATO-TS), using a test equipment named ATO-TE (Test Equipment)

5.3.1 Phasing

At this stage, two phases are considered for Alstom ATO:

- Phase 1: application in Flat-IP only
    - Pre-integration performed in Alstom labs beginning in August 2021.
    - Integration in Kontron labs (WP4) in November 2021
- Phase 2: Loose couple approach, OBAPP implementation
    - Pre-integration in Alstom labs beginning in January 2022
    - Integration in Kontron labs (WP4) in March 2022

Hence, only phase 1 is considered for the first delivery to WP4 in November 2021.

5.3.2 Additional pre-integration tests – Phase 1 (Flat IP)

Tests in nominal conditions:

These tests are described in chapter 8.3.1 of D1.1, the aim is to check the behaviour under nominal condition. Some of these tests are executed during pre-integration stage, as explained below:

DESCRIPTION OF THE TESTS TO BE DONE IN WP4	CORRESPONDING D1.1 CHAPTER	STATUS OF THE CORRESPONDING PRE-INTEGRATION TEST IN ALSTOM LABS
Establish a communication between ATO-OB and ATO-TS. Check that communication is correctly set with PROBE tool (ATO test and diagnostic tool). The test scenario is managed by ATO-TE equipment and is described below this table.	8.3.1.3	It is fully performed during pre-integration tests.

Table 57 Description of the tests in nominal conditions to be done in WP4

Detailed test description:

1. Launch all softs: by opening the cmd files Start_5Grail_OB and Start_5Grail_TE
2. Go to the application ATO_REPLAY, and load the scenario TEST_5G.STrm then launch it
3. Check in the PROBE, ATO_trackside and SS126 the Handshake request issued by the ATO-onboard
4. Check in the ATO-trackside the content of the received Handshake request
5. Check the PROBE, ATO_trackside and SS126 the handshake Acknowledgement issued by the ATO-trackside
6. Check at ATO-onboard level if the handshake Acknowledgement is coherent with the message sent by the ATO-trackside
7. Check the PROBE, ATO_trackside and SS126 the Journey Profile request issued by the ATO-onboard
8. Check the ATO-trackside and PROBE the content of the received Journey profile request
9. Check the PROBE and ATO_trackside the journey profile issued by the ATO-trackside
10. Check at ATO-onboard level if the Journey profile is coherent with the message sent by the ATO-trackside
11. Check in the PROBE the segment profile request issued by the ATO-onboard
12. Check in the ATO-trackside the content of the received segment profile request
13. Check in the PROBE, ATO_trackside and SS126 the segment profile issued by the ATO-trackside
14. Check at ATO-onboard level if the segment profile is coherent with the message sent by the ATO-trackside
15. Check in the PROBE the status report issued by the ATO-onboard
16. Check in the ATO-trackside the content of the received status report
17. Check in the PROBE the status report acknowledgment issued by the ATO-trackside
18. Check at ATO-onboard level status report acknowledgment is coherent with the message sent by the ATO-trackside

Tests in degraded conditions:

These tests are described in chapter 8.3.2 of D1.1, the aim is to check the behaviour under some degraded scenarios. Some of these tests are executed during pre-integration stage, as explained below:

DESCRIPTION OF THE TESTS TO BE DONE IN WP4	CORRESPONDING D1.1 CHAPTER	STATUS OF THE CORRESPONDING PRE-INTEGRATION TEST IN ALSTOM LABS
Not finalized yet in D1.1 deliverable	8.3.2.3	After a correct communication establishment between OB and TS, trigger a mobile switch on a second radio bearer. Check that the data transfer is still on-going and the communication is not impacted by the switch. The nominal communication is on 5G channel. We use attenuator to simulate a loss of 5G network, and switch to a Wi-Fi or 4G link. Same actions than tests in nominal conditions, the logs between nominal and degraded conditions shall be compared.

Table 58 Description of the tests in degraded conditions to be done in WP4

5.3.3 Additional pre-integration tests – Phase 2 (Loose-coupling)

The pre-integration tests performed for phase 2 are the one described in chapter **4** for OB_{APP}/TS_{APP} API for the control plane. Once the control plane allows to establish a session between ATO-OB and ATO-TS, the same user plane test for nominal condition (see chapter **5.3.2**) was successfully performed.

5.4 ETCS/ATP application (CAF)

No additional tests (appart from the chapter 3 test cases) are planned.

5.5 CCTV Offload/Video (Teleste)

No additional tests (appart from the chapter 3 test cases) are planned.

5.6 PIS (Thales)

The purpose of this chapter is to describe additional PIS pre-integration test cases.

5.6.1 Phasing plan

PIS lab is located on two different geographical sites:

1. In Kontron's premises at Montigny-Le-Bretonneux, France.
2. In Thales SIX-GTS' premises at Vélizy, France.

Figure 4: Geographical situation of PIS Lab.

In Kontron's premises are installed, among others:

- 5G Radio Access Network and Core Network,
- IMS and MCx systems,
- FRMCS On-Board and TrackSide Gateways,
- VPN Gateway for the remote access with Thales Lab.

In Thales SIX-GTS' premises are installed:

- PIS TrackSide and On-Board equipment,
- VPN Gateway for the remote access with Kontron Lab

Note: The description of the different equipment is given in Deliverable D4.1.

In order to be able to execute functional PIS test cases (i.e., send trackside to on-board and on-board to trackside messages), a secure VPN connection must be considered between the two sites to interconnect PIS equipment to the FRMCS infrastructure.

At this stage, two phases are considered for Thales-PIS:

- Phase 1: PIS in flat-IP mode
    - Remote connection pre-integration beginning in June 2021
    - PIS pre-integration in October 2021
    - PIS integration in November 2021
- Phase 2: PIS in loose coupled mode, OBapp and TSapp implementation
    - PIS pre-integration in January 2022
    - PIS integration in May 2022

5.6.2 Additional pre-integration tests in Phase 1

5.6.2.1 Remote connection

Figure 5 illustrates the network architecture the PIS lab to validate pre-integration tests of the secured remote connection between the labs of Thales and Kontron.

During this phase, the aim is to validate the remote connection configuration so to simplify the configuration the PIS trackside and on-board equipment is replaced by laptops as well as the FRMCS trackside and on-board Gateways.

- PIS trackside equipment is replaced by a laptop named “LAPTOP_OCC”;
- PIS on-board equipment is replaced by a laptop named “LAPTOP_TRAIN”;
- FRMCS trackside gateway is replaced by the laptop named “LAPTOP_TS_GW”;
- FRMCS on-board gateway is replaced by the laptop named “LAPTOP_OB_GW”;

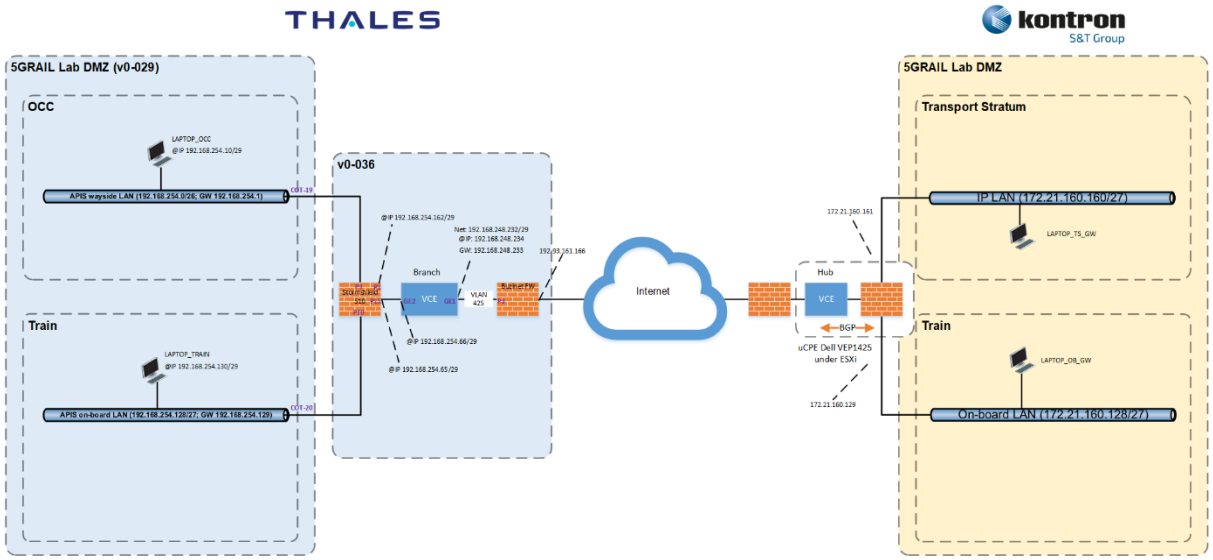

Figure 5: PIS lab architecture for pre-integration tests in phase 1 for the remote connection.

Table 59 provides the IP addresses of the different equipment involved in the pre-integration tests of the remote connection in phase 1.

EQUIPMENT	IP ADDRESS
LAPTOP_OCC	192.168.254.10/26
LAPTOP_TRAIN	192.168.254.130/27
LAPTOP_TS_GW	172.21.160.162/27
LAPTOP_OB_GW	172.21.160.130/27

Table 59 IP plan used for the pre-integration tests of the remote connection in Phase 1.

5.6.2.1.1 TC_001: REACHABILITY OF TRACKSIDE EQUIPMENT

5.6.2.1.1.1 PURPOSE

The purpose of these tests is to validate the ability of the PIS trackside equipment to reach the FRMCS trackside Gateway and the FRMCS trackside Gateway to reach the PIS trackside equipment.

5.6.2.1.1.2 DESCRIPTION OF THE INITIAL STATE/CONFIGURATION

The initial state covers the following steps:

- Laptops are configured and up,
- Kontron VCE are configured and up,
- Thales and Kontron VPN gateways are configured to authorize ICMP packets between the different equipment,
- The internal firewall of the laptops is deactivated,
- Network protocol analyser (e.g., Wireshark) is started and configured to capture traffic on the network interface involved in the tests.

5.6.2.1.1.3 TEST PROCEDURE 1: PIS TRACKSIDE EQUIPMENT REACHES FRMCS TRACKSIDE GATEWAY

STEP	ACTION	EXPECTED RESULT(S)
1	Execute “ping” command to send ICMP ECHO-REQUEST messages from LAPTOP_OCC to the IP address of LAPTOP_TS_GW.	LAPTOP_OCC shall receive ICMP ECHO-REPLY messages from LAPTOP_TS_GW.

Table 60 test procedure 1: PIS trackside equipment reaches FRMCS trackside gateway

5.6.2.1.1.4 TEST PROCEDURE 2: FRMCS TRACKSIDE GATEWAY REACHES PIS TRACKSIDE EQUIPMENT

STEP	ACTION	EXPECTED RESULT(S)
1	Execute “ping” command to send ICMP ECHO-REQUEST messages from LAPTOP_TS_GW to the IP address of LAPTOP_OCC.	LAPTOP_TS_GW shall receive ICMP ECHO-REPLY messages from LAPTOP_OCC.

Table 61 test procedure 2: FRMCS trackside gateway reaches PIS trackside equipment

5.6.2.1.2 TC_002: REACHABILITY OF ON-BOARD EQUIPMENT

5.6.2.1.2.1 PURPOSE

The purpose of these tests is to validate the ability of the PIS on-board equipment to reach the FRMCS on-board Gateway and the FRMCS on-board Gateway to reach the PIS on-board equipment.

5.6.2.1.2.2 DESCRIPTION OF THE INITIAL STATE/CONFIGURATION

The initial state covers the following steps:

- Laptops are configured and up,
- Kontron VCE are configured and up,
- Thales and Kontron VPN gateways are configured to authorize ICMP packets between the different equipment,
- The internal firewall of the laptops is deactivated,
- Network protocol analyser (e.g., Wireshark) is started and configured to capture traffic on the network interface involved in the tests.

5.6.2.1.2.3 TEST PROCEDURE 1: PIS ONBOARD EQUIPMENT REACHES FRMCS ON-BOARD GATEWAY

STEP	ACTION	EXPECTED RESULT(S)
1	Execute “ping” command to send ICMP ECHO-REQUEST messages from LAPTOP_TRAIN to the IP address of LAPTOP_OB_GW.	LAPTOP_TRAIN shall receive ICMP ECHO-REPLY messages from LAPTOP_OB_GW.

5.6.2.1.2.4 TEST PROCEDURE 2: FRMCS ONBOARD GATEWAY REACHES PIS ONBOARD EQUIPMENT

STEP	ACTION	EXPECTED RESULT(S)
1	Execute “ping” command to send ICMP ECHO-REQUEST messages from LAPTOP_OB_GW to the IP address of LAPTOP_TRAIN.	LAPTOP_OB_GW shall receive ICMP ECHO-REPLY messages from LAPTOP_TRAIN.

5.6.2.2 PIS communication

5.6.2.2.1 TC_001: SEND TEXT MESSAGE WITH A NORMAL PRIORITY TO TRAINS IN NORMAL CONDITIONS

This Test Case is described in chapter 9.6.1.2 of D1.1 v1.

5.6.2.2.2 TC_002: ON-BOARD PIS LOGS DOWNLOADED ON THE FLY IN NORMAL CONDITIONS

This Test Case is described in chapter 9.6.1.6 of D1.1 v1 except here we do not expect to degrade 5G radio link.

5.6.3 Additional pre-integration tests in Phase 2

For the first WP4 delivery in November 2021, the pre-integration tests related to phase 2 will not be performed.

A revision could be done for the second delivery to WP4 to include Phase 2 tests.

5.7 Siemens

No additional tests (appart from the chapter 3 test cases) are planned.

6 CONCLUSIONS

The sudden appearance of Covid-19 had an impact on every project, but especially on projects like 5Grail that included prototype integration test that required the presence of people on-site. This risk was present from almost the beginning of the project, and therefore, a mitigation plan was elaborated to minimize the impact on the project planning.

The pre-integration tests, as well as the lab test activities had to be designed to be performed remotely. The pre-integration tests were executed locally by each application supplier, using the FRMCS GW simulators provided by the GW suppliers. Thus, the application suppliers were able to test and debug the interface between the applications and the GWs.

The pre-integration studies as described in this document can be considered a success in the light of the difficulties due to the need to perform integration and testing remotely. All in all, Covid-19 and the measures taken to contain the virus took a toll on the performance. The mitigation plan put in place to minimize the impact worked, within limits.

The framework provided by the industry participants is working and already enhancing the integration while reducing time and effort needed in the lab and – ultimately – at the test tracks in France and Germany.

During the pre-integration phase, testing included relevant protocol stacks and messages for ATO, ETCS, TCMS, PIS, CCTV Offload/CCTV Live in the provided (and somewhat limited) environment.

This document provides testimony to the successful communications between the interfaces OBapp and TSapp to the OB Gateway for various applications. This will allow a more rapid refinement and further development of FRMCS in the work packages WP3 and WP4 in France and Hungary. Additionally, both field test environments will benefit from these results.

Finally, this documents can be used in the future for any other onboard applications to stress its compliancy towards OBapp and TSapp interface.

7 REFERENCES

Deliverable D1.1 - Test Plan (Work Package 1)

Deliverable D2.1 - Architecture Report (Work Package 2)

Deliverable D3.1 - First Lab Integration and Architecture Report (Work Package 3)

Deliverable D4.1 - Second Lab Integration and Architecture Report (Work Package 4)

Grant agreement
No 951725