Innovatief Partnership

Talking Traffic

Software Test Description RIS

Cluster 1

Date	*22-02-2018*
Version	*2.0*
Company	dynniq energising mobility
Cluster	*1*
Work package	*1*

1 Voorwoord

Binnen het Innovatiepartnership Talking Traffic (🕮 11) wordt in Cluster 1 het RIS FI koppelvlak gebouwd, in de praktijk beproefd en wordt de goede functionele en technisch werking aangetoond. Ook dient de coöperatieve data die de RIS-faciliteit in of aan de VRI ontvangt van voertuigen, doorgeleverd aan Cluster 2.

In het Software Test Plan van Cluster 1 (🕮 6) wordt de test strategie beschreven voor de werk pakketten 1, 2 en 3 binnen Talking Traffic. Dit document beschrijft de FAT test specificatie voor de RIS, welke daar onderdeel van is.

Dit document is tot stand gekomen door samenwerking van de vijf leveranciers in de werkgroep bestaande uit:

dynniq energising mobility

Royal HaskoningDHV Enhancing Society Together

swarco

Vialis

SWECO

NB. De rest van dit document is geschreven in het Engels om internationale uitwisseling te ondersteunen.

The rest of this deliverable has been written in English to facilitate international exchange.

DOCUMENT CONTROL SHEET

Document Versions

Version	Date	Author	Comments	Checked / Approved by
1.0	21-12-2016	Erik Hendriksen	Initial RIS-FI test document	
1.1	22-05-2017	Robert Eveleens	Reformatting in TT style, extending tests, adding RIS cases additional requirements	
1.2	26-06-2017	Robert Eveleens	Processed review comments Added TLEX tests, compliance paragraph and test overview appendix	
1.3	6-07-2017	Robert Eveleens	Processed review, final release	
1.4	7-07-2017	Robert Eveleens	Missed review comment, updated final release	
1.5	18-07-2017	Robert Eveleens	Rework from comment I&M	
1.6	13-09-2017	Robert Eveleens	Rework to fit STP 1.3, multiple comments and minor typo's	
1.6.1	16-10-2017	Robert Eveleens	Updated reference 4 Updated Test events to fit TINT1 configuration	
1.6.2	30-11-2017	Erik Hendriksen Reynier Braal	Minor updates MAP, SPAT and SSM messages Added appendix C, D and E	
1.7	22-12-2017	Erik Hendriksen	Draft version for review for RIS-FI 1.2	
1.8	12-02-2018	Erik Hendriksen	Draft version for review. Processed review results of 1.7	
1.9	19-02-2018	Erik Hendriksen	Draft version for approval Processed review results of 1.8	
2.0	22-02-2018	Erik Hendriksen	Final version for RIS-FI 1.2	

CONTENT

2 Introduction

This document describes the Software Test Description (STD) for the type test of the Roadside ITS Station (RIS) in order to verify the compliance of the RIS under test to the requirements referenced in *§3.1 Normative.*

2.1 System overview

The iTLC architecture as shown in Figure 1 combines the ability to control traffic lights and the ability to communicate to ITS stations such as cars, busses etc.

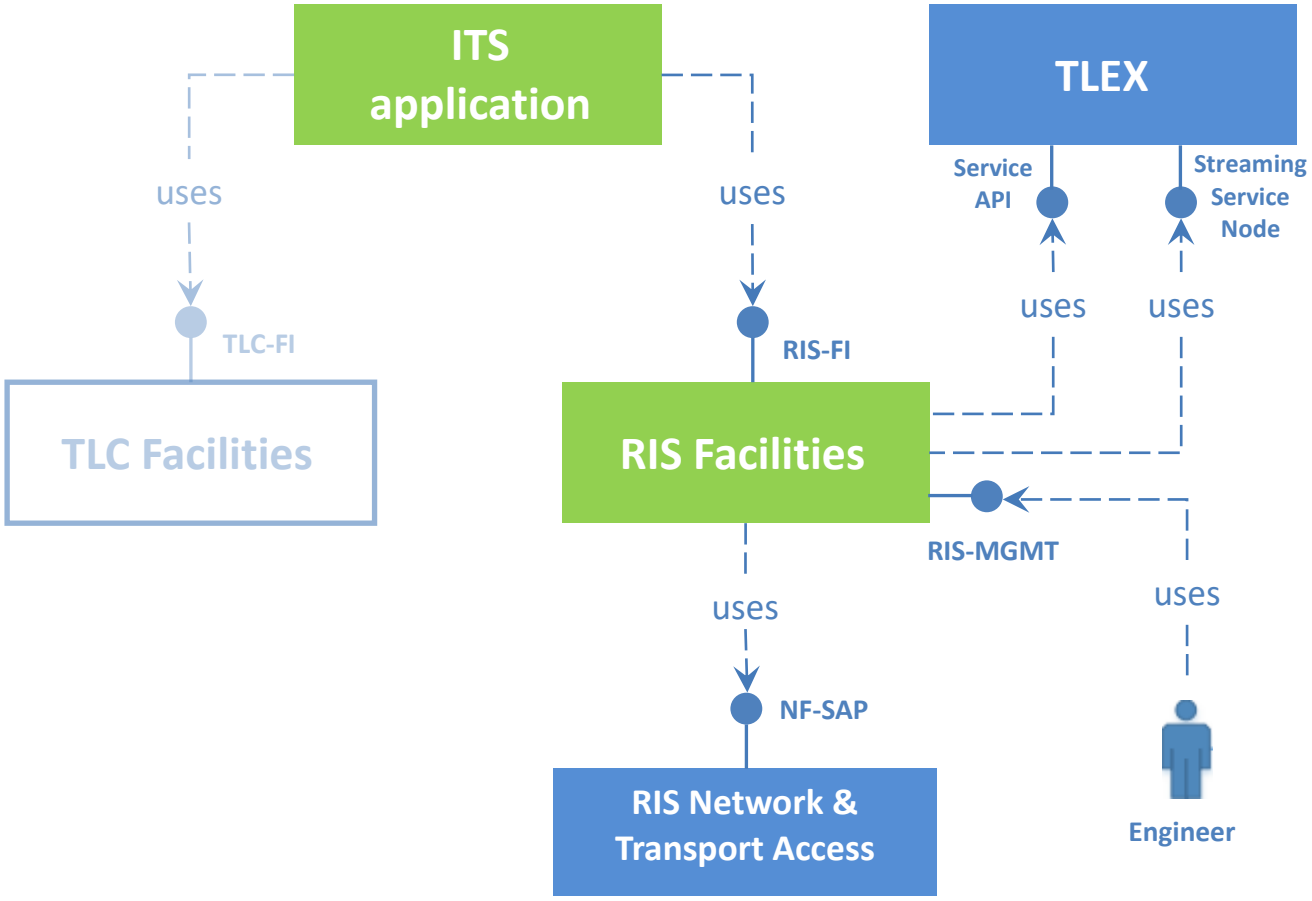

Figure 1: RIS in the iTLC system overview

It allows (external) ITS applications to control or monitor traffic lights via the TLC-FI interface. It also allows ITS applications to monitor or inform ITS stations via the RIS-FI interface. See 🕮 5 for an overview of the iTLC architecture.

The RIS can communicate with other ITS stations in the neighborhood via C-ITS messages. The information received from ITS stations and the information received from ITS applications via the RIS-FI, is used to assemble a local view on the traffic situation.

Real time data from and to the iTLC is communicated through the RIS to TLEX data exchange point (overnamepunt), which enables data exchange between cluster 1 and cluster 2. See 🕮 4 for a description of the exchange point.

2.2 Scope

The scope of this document is limited to the RIS Facilities and the highlighted parts in Figure 1. The following is covered in the tests described in this document:

- The RIS functionality
- The RIS-FI interface
- The RIS-MGMT interface
- The TLEX interface (4G/LTE, connected ITS)
- The NF-SAP interface (Wifi-p, cooperative ITS)

The NF-SAP interface is not mandatory for the Talking Traffic partnership and is provided for hybrid RIS developers.

Although the RIS Management interface (RIS-MGMT) is mandatory for the RIS, it is not specified and the interpretation can differ per vendor. The information for this interface has to be provided by the vendor.

2.3 Document overview.

2.3.1 Purpose

This document provides specifications for the type testing of the RIS. This document is a part of a set of test specifications for testing the complete iTLC Architecture as described in 🕮 11 *RFP Talking Traffic 1.1 Beter Benutten*.

The suite of tests for the cluster one tests of TT is defined in the 🕮 6 *IPS-TT STP Cluster 1***.** The document describes which software systems in cluster 1 are tested and to which document the systems have to comply.

Note: As the requirements are not all final and the development is still ongoing this document is subject to change. Therefore assure the last version is used during testing.

2.3.2 Document Structure

Following is the document structure:

Chapter 3 contains references to normative and informative documents

Chapter 4 explains acronyms and used definitions and concepts

Chapter 5 outlines the required test setup

Chapter 6 outlines guidelines for the execution of the test scenarios and the structure of the tests

Chapter 7 outlines the formal specification of the test cases

Appendix A contains the requirements traceability

Appendix B contains an overview for entering the test results

Appendix C contains a complete MAPData message in JSON format

Appendix D contains a complete SPAT message in JSON format

Appendix E contains a complete SSM message in JSON format

2.4 Advise for reader

The reader should be acquainted with the normative requirements (*§3.1*) before reading the test specification.

3 References

3.1 Normative

Ref.	Specification	Version	Date/Year
🕮 1	Beter Benutten Vervolg, project iVRI, Deliverable G1: IRS RIS-FI	1.2	27-01-2016
🕮 2	Beter Benutten Vervolg, project iVRI, Deliverable 1ab: IDD Generic-FI	1.1	02-12-2016
🕮 3	Beter Benutten Vervolg, project iVRI, Deliverable 1b: IDD RIS-FI	1.2	01-12-2017
🕮 5	Beter Benutten Vervolg, project iVRI, Deliverable F, iTLC Architecture	1.2	27-01-2016
🕮 6	IPS-TT STP Cluster 1	1.3	12-09-2017
🕮 8	Dutch Profile Intersection Topology Format	1.2	20-06-2017
🕮 9	IRS Security v1.1	1.1	13-10-2016
🕮 10	Dutch Profile CAM Profile	1.2	29-06-2017
🕮 11	Dutch Profile MAP Profile	1.2	29-06-2017
🕮 13	Dutch Profile SRM Profile	1.2	29-06-2017
🕮 14	Dutch Profile SSM	1.2	29-06-2017
🕮 15	Dutch Profile SPAT	2.0	16-11-2017

3.2 Informative

Ref.	Specification	Version	Date/Year
🕮 10	Beter Benutten Vervolg, project iVRI, Deliverable 3f: Test strategy	1.2	6-12-2016
🕮 11	RFP Talking Traffic 1.1 Beter Benutten	1.1	1-07-2016
🕮 12	Beter Benutten Vervolg, project iVRI – fase 2, Deliverable 3f iVRI test strategy	1.2	06-12-2016
🕮 13	ETSI TS 102 894-2 v1.2.1	1.2.1	2014-09

4 Acronyms and abbreviations

Item	Description
CAM	Cooperative Awareness Message
BBV	Beter Benutten Vervolg: Program for standardization of interfaces with TLCs for connected and cooperative functionality
DENM	Decentralized Environmental Notification Message.
ETSI	European Telecommunications Standards Institute
IDD	Interface Design Description.
IRS	Interface Requirements Specification
iTLC	Intelligent TLC performing traffic light controller functions and allowing for ITS applications.
IOP Test	Interoperability Test
C-ITS	Cooperative ITS functionality for exchange of data between in-vehicle and or road side devices making use of either cellular or short range wireless communication
iTLC (Dutch iVRI)	Intelligent TLC performing traffic light controller and C-ITS functions and providing access to these functions for ITS applications
ITS	Intelligent Transport Systems.
ITS G5	ITS messages broadcasted over the 5GHz radio band supporting GeoNetworking, as specified by ETSI.
MAP	Message to convey the current road topology to road-users, often used in conjunction with SPAT
OWASP	Non-profit organisation dedicated to secure software development.
RIS	Roadside ITS Station
RIS-FI	RIS Facilities Interface
RIS-MGMT	RIS Management Interface
SPAT	Signal Phase and Timing (message providing traffic light information).
TLS	Transport Layer Security
TTCN-3	Testing and Test Control Notation Version 3
ITS Station	Functional entity specified by the ITS station reference architecture (see ETSI EN 302 665, V1.1.1)
ITS-A	ITS Application
ITS-CLA	ITS Control Application
ITS-CRA	ITS Consumer Application
ITS-PRA	ITS Provider Application
TLEX	Traffic Light EXchange platform, broker for exchanging C-ITS messages between cluster one and two
Talking Traffic	Partnership of service providers within the framework of the Call, aimed at the development and operation of Services to implement Use Cases
SSM	Signal Status Message; the state of a priority request.
SRM	Signal Request Message; a priority request.
JSON	JavaScript Object Notation
Wireshark	Network protocol analyser tool
DSRC	Dedicated Short Range Communication message set (defined in SAE J2735:2016)
GeoNetworking	Network layer protocol that provides packet routing in an ad hoc network based on geographic location, used in the IEEE 802.11p
IEEE 802.11p	IEEE standard for adding wireless access in vehicular environments to the WIFI protocol (WIFI-P), base of the ETSI ITS-G5
WIFI-P	The IEEE 802.11p protocol
LDM	Local Dynamic Map, holds the overall view on the traffic state in the area that the Roadside ITS Station (RIS) covers
NMAP	Network Mapper, is an open source tool for network exploration and security auditing.

5 Test Setup

5.1 Introduction.

This section describes the test setup and is an advisory on how to setup a test environment

5.2 Test setup

The device under test (DUT) is a RIS that implements the requirements of *§3.1 Normative* and is subjected to the type testing. The following figure gives an overview of the required test setup.

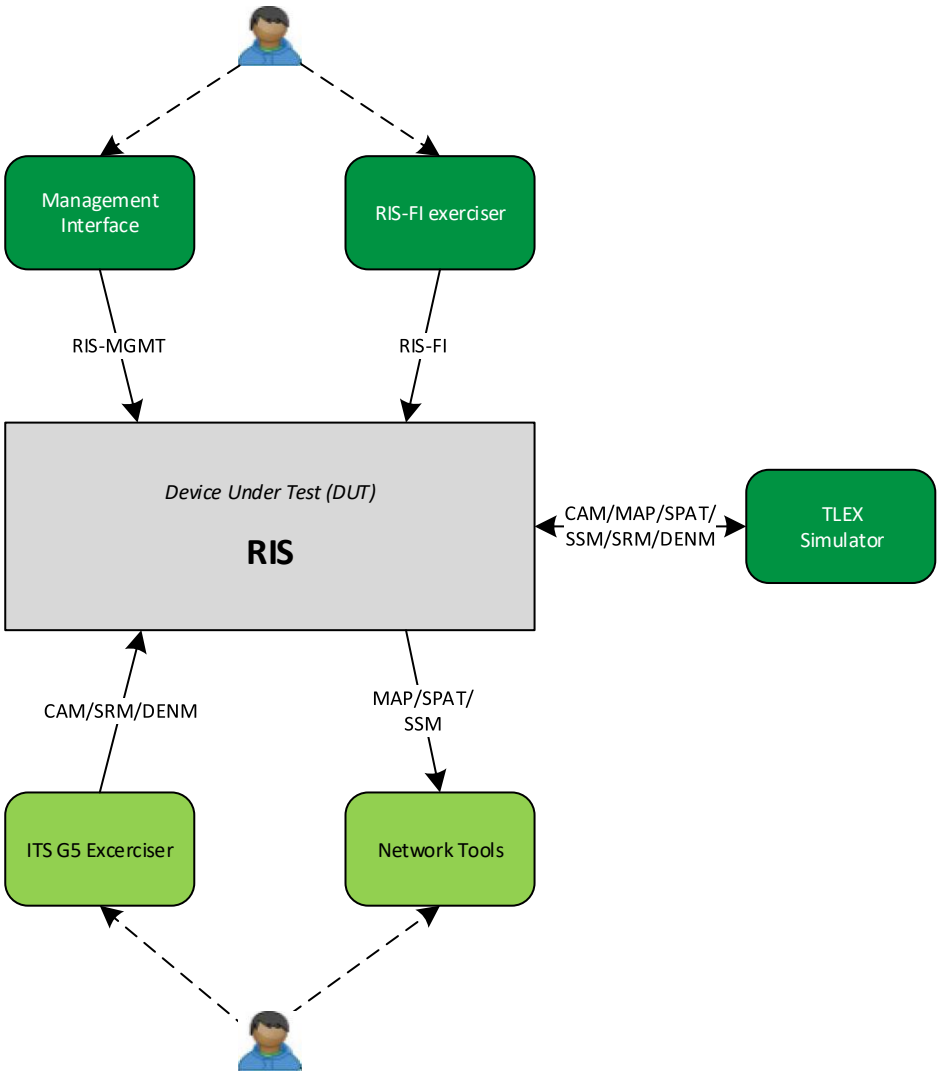

Figure 2: Test setup

The RIS is considered as a black box exchanging C-ITS messages from ITS stations and the exchange point (TLEX) and exchanging objects created or updated by ITS applications via the RIS-FI.

For testing session disconnection during (soft) reset it must be possible to do an actual (soft) reset of the RIS.

Prior to the test, TLS certificates need to be available and deployed on the RIS.

5.2.1 RIS-FI exerciser

For testing the RIS-FI interface a RIS-FI Exerciser is required. This is a non-standard software component able to communicate via the RIS-Interface and has functionality to test the interface.

The RIS-FI exerciser can:

- connect to a RIS using the RIS-FI interface
- maintain a connection to the RIS (implements the required state machine)
- connect using TLS 1.2
- update the RIS-FI protocol version in JSON message for testing version control
- send a customized JSON message
- implement the full RIS-FI
- revoke the session with the RIS.
- enable / disable the keep alive mechanism

Optional the RIS-FI Exerciser can:

- show and / or log received RIS-FI messages
- The RIS-FI exerciser will act as one of the following ITS Application (ITS-A) types:
    - ITS-CRA, ITS Consumer Application
    - ITS-PRA, ITS Provider Application

5.2.2 TLEX simulator

The TLEX simulator implements the TLEX TCP streaming protocol (📖 4).

The TLEX simulator is used to generate CAM and SRM messages.

The TLEX simulator records/logs all messages received from the iTLC.

The TLEX simulator is NTP time synchronised. The TLEX simulator timestamps all messages.

5.2.3 Management Interface

As mentioned in (📖 5) the RIS-MGMT (Roadside ITS Station Management) interface is mandatory for the RIS, but not specified. The vendor has to provide the details of the interface and the required access codes.

The RIS maintains several log files containing errors and significant events about the operation of the RIS. At least the following is logged:

- registration attempts and result
- role switches
- SPAT-performance (SPAT data availability)
- information about transmitted and received ITS-G5 messages (may contain message content as well)

These log files can be accessed by using the Management Entity.

5.2.4 ITS G5 exerciser

The ITS G5 exerciser is capable of sending CAM, SRM and DENM messages to the RIS in order to emulate the NF-SAP shown in Figure 1: RIS in the iTLC system overview.

The ITS G5 exerciser can:

- Generate CAM, SRM and DENM messages

Optional the ITS G5 exerciser can:

- Log exchanged messages

5.2.5 Network tools

Monitoring of the messages transmitted by the RIS (MAP, SPAT and SSM) can be done by using a network protocol analyzer like Wireshark for which plugins are available to decode the C-ITS messages.

The following tools are used for verifying network traffic and security auditing

Tool	URL	Description
Wireshark	https://www.wireshark.org/	Network protocol analyser
ETSI plugins for Wireshark	http://oldforge.etsi.org/websvn/listing.php?repname=ITS.WIRESHARK_ITS_PLUGINS	ITS G5 message decoder plugins
OWASP o-saft tool	https://www.owasp.org/index.php/O-Saft	Advanced forensic tool / OWASP SSL audit for testers
nmap	https://nmap.org/	Nmap ("Network Mapper") is an open source tool for network exploration and security auditing.

Table 1: Network tools

5.2.6 Test Automation Tools

With the use of automated testing tools test cases can be executed in a predefined timely matter. It is also more convenient to test exceptions and timing behavior.

The ETSI provides ITS tests for the GeoNetworking (ITS-G5), described in the TTCN-3 language (Testing and Test Control Notation Version 3), available at the following location:

http://forge.etsi.org/websvn/listing.php?repname=ITS.ITS&

This can be used as a base for creating automated tests in which the RIS-FI exerciser, ITS G5 exerciser and the TLEX simulator functionality can be combined.

Especially the tests described in 7.3.5 Additional Case: Performance tests can best be done using automated tests.

Note: Multiple tests are available where objects have a lifetime which is of effect for the next step of the test case. These can be best done using automated testing. If no automated testing is done, assure, prior to starting the test, that the test can be performed within the configured time frame.

5.2.7 Additional tools

For testing if the RIS takes appropriate action when an incompatible protocol is trying to connect (see 7.3.4.8) an Ivera client is used.

5.3 Test configuration

The test configuration is specified in Appendix A of the STP (🕮 6).

5.3.1 Test events

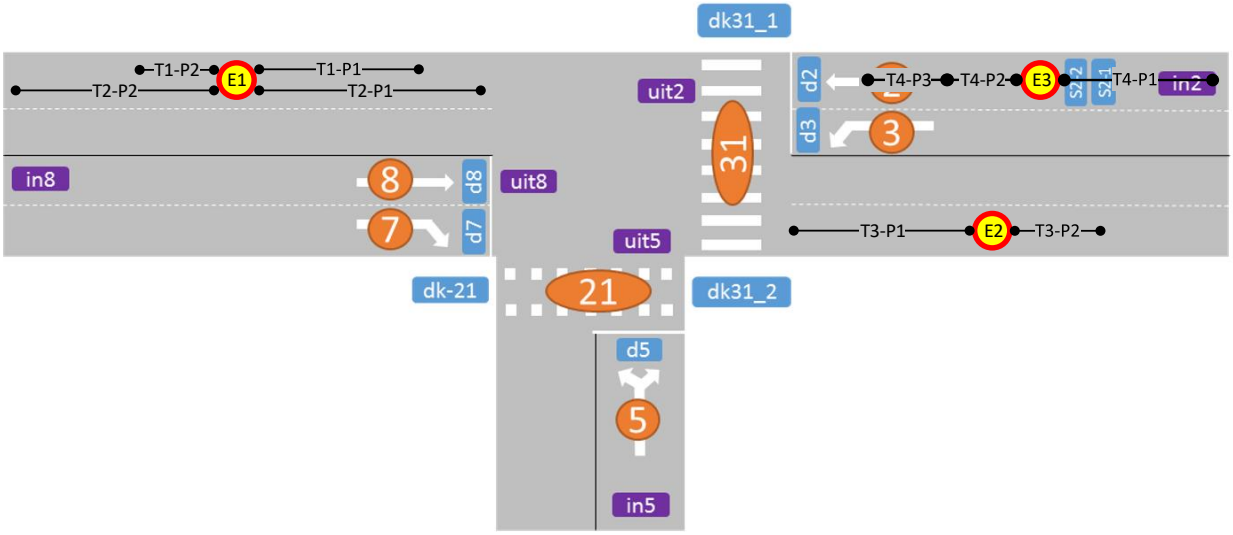

Figure 3: Test intersection topology

For testing of event positions the following are allocated on the intersection as shown in the figure above (not on scale):

Reference	Type	Description
E1	eventPosition	E1 is 50 meter to the left from the intersection center on path 10, indicated in Figure 3: Test intersection
T1	trace	T1 contains two paths: P1 starts 30 meters before E1 until E1 P2 starts from E1 until 10 meters after E1
T2	trace	T2 contains two paths: P1 start 50 meters before E1 until E1 P2 start from E1 until 50 meters after E1
E2	eventPosition	E2 is 20 meter to the right from the intersection center on path 4 indicated on Figure 3: Test intersection
T3	trace	T3 contains two paths: P1 start 70 meters before E2 until E2 P2 start from E2 until 30 meters after E2
E3	eventPosition	E3 is 100 meter to the right from the intersection center on path 1 indicated on Figure 3: Test intersection
T4	trace	T4 contains three paths: P1 start 150 meters before E3 until E3 P2 start from E3 until 50 meters after E3

		P3 start from P2 until 20 meters after P2

Table 2: Events reference

For mapping vehicles on the topology absolute GPS coordinates are required and will be provided in the topology file.

Note: WGS84 coordinates will be provided after the configuration is available.

5.3.2 Default Registrationrequest

Following are the default **RegistrationRequest** fields if not provided in the test:

version := {1,2,0},

uri := "http://www.<your_company>/tester" or http://www.<ip>:<port>

5.3.3 Default network ports

The default network ports of the RIS are given in the table below.

Facilities	Port
RIS Facilities (TLS)	12001
RIS Facilities (no security)	12501

Table 3: Default network protocol ports

These are not mandatory and can differ from the actual implementation. If ports other than default are used, note them in the test results.

5.3.4 Messages defaults

For generating CAM messages the following default values will be used, unless stated otherwise in the test case (units as defined in 📖13):

generationDeltaTime = calculated from current system time
stationType = StationType_passengerCar_(5)
referencePosition = TINT1 Intersection reference position.
altitude.altitudeValue = unavailable(800001)
heading:
- headingValue = 2700 (270 degrees)
- headingConfidence = unavailable(127)

speed:
- speedValue =1250 (12,5 m/s)
- speedConfidence = unavailable(127)

driveDirection = forward(0)
vehicleLength:
- vehicleLengthValue =50 (5 meter),
- vehicleLengthConfidenceIndication = noTrailerPresent (0)

vehicleWidth = 20 (2 meter)
longitudinalAcceleration:
- longitudinalAccelerationValue = unavailable (161)
- longitudinalAccelerationConfidence = unavailable (102)

curvature:
- curvatureValue = unavailable(30001)
- curvatureConfidence = unavailable (7)

curvatureCalculationMode = unavailable(2)
yawRate:
- yawRateValue = unavailable(32767)
- yawRateConfidence = unavailable (8)

vehicleRole = default(0)
exteriorLights = 00000000
pathHistory[0] = referencePosition

Table 4 : Default CAM values

For generating SRM messages the following default values will be used, unless stated otherwise in the test case (units as defined in 📖13):

timeStamp = calculated from current system time
second = calculated from current system time
sequenceNumber = 1
requests[0]:
- request
  - id.region = TINT1 Intersection region
  - id.id = TINT1 Intersection id
  - requestID = 100
  - requestType = priorityRequest(1)
  - inboundLane = { lane = 1 }
- minute = calculated from current system time + 10s
- second = calculated from current system time + 10s
- duration = 0

requestor:
- id.stationID = identical to stationID of the CAM
- type.role = basicVehicle(0)
- type.subrole = requestSubRoleUnKnown(0)

Table 5 SRM defaults

For all C-ITS messages goes that, if not specified in the test, the mandatory fields have to be added and default values have to be provided.

6 Test execution

6.1 Introduction

This chapter provides guidelines for the execution of the test scenarios specified in this document.

6.2 Structure

During a test the Device Under Test (DUT) is subjected to the documented test cases. The tests are organized by interface and use case as follows:

- §7 Test scenarios
    - Interface
        - Use Case(s)
            - Test case(s)
                - Test step

Procedures already tested in a use case, are assumed to be known and functional and can be used as a pre-condition of a test case in another use case.

The following test scenarios are defined under test scenarios:

Interface	Paragraph	Description
RIS-FI	§ 7.2	The functional use cases described in 🕮 3 §8 are used. Optional test cases are added to verify functionality not covered in the use cases, but which are closely related to the topic. Also, basic Generic-FI tests are covered to provide a logical build up. As an additional use case the **RequestObject** method is added as this is not handled in the defined use cases. The first use case *7.2.1 Use Case: Monitoring of traffic* is a happy flow sequence where the basic functionality is tested. It is advised to start testing with this use case.
Generic-FI	§ 7.3	The RIS-FI is built upon the Generic-FI. Functionality from the generic-FI, not already covered in the RIS-FI tests, are handled in this section. The functional use cases are taken from 🕮 2 §8. Additional Performance and security tests are added.

Table 6 : Test scenarios overview

Where possible, test steps are added for verifying compliance to:

- 🕮 5 iTLC Architecture
- 🕮 10 IRS security

6.3 Assumptions and constraints

Most of the tests are done with a limited number of objects in the LDM of the RIS. It is not necessary to have a large number of objects in the LDM for testing the RIS-FI interface object content. Performance tests are designed to use the maximum number of entities as described in the IRS.

6.4 Execution

The tests are executed in the documented order. No alternations should be made to the device under test (the RIS) during the test, unless explicitly documented.

The results are documented per test step.

A test step has passed if the pass criteria has been met.
A test step has failed if the pass criteria has not been met.
A test case has passed if all test steps have passed.
A test scenario has passed if all test cases have passed.

6.5 Test case notation format

The following format is used to specify the test cases and document the test results.

Test case:			
ID:			
Objective:			
Pre-conditions			
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1			
2			
3			
4			
Tested by:		**Date:**	

- Test case : A short description of the test case.
- ID : A unique ID
- Objective : The objective of the test case.
- Pre-conditions : The pre-conditions before the test case is executed.
- STEP : Number of the step.
- DESCRIPTION : A description of the actions to be executed and a description of the items to be verified.
- PASS/FAIL : The result of the test is either PASS or FAIL
- REMARKS/ACTIONS : Remarks and actions related to the test results. In case of a FAIL the actual result can be logged as also the registration number of the issue that is discovered & logged during the test.
- Tested by : Person who executed the test case.
- Date : The date at which the test case was executed.

6.6 Conventions

Throughout the test case different text styles are used.

Actions are presented in bold and underscored as follows:

Verify an empty result is returned with a subscription id, different than the previous one

Note the received **subscription** ID: _____

Object names are presented in bold and shown in the font that is shown in the following example:

The **VehicleRole** is **publicTransport**

6.7 Compliance

A RIS can use the following methods for exchanging C-ITS messages:

- Connected, using 4G/LTE and the TLEX
- Cooperative, using the NF-SAP
- Hybrid, using both interfaces

Note in the test registration (7.1) which compliance is tested.

6.7.1 Connected

For the “Beter Benutten” project the RIS has to comply with the connected method for exchanging messages. In this case, all tests have to be done using the TLEX Simulator.

The C-ITS messages sent by the RIS are verified at TLEX Simulator and messages sent by an ITS Station are sent by the TLEX Simulator.

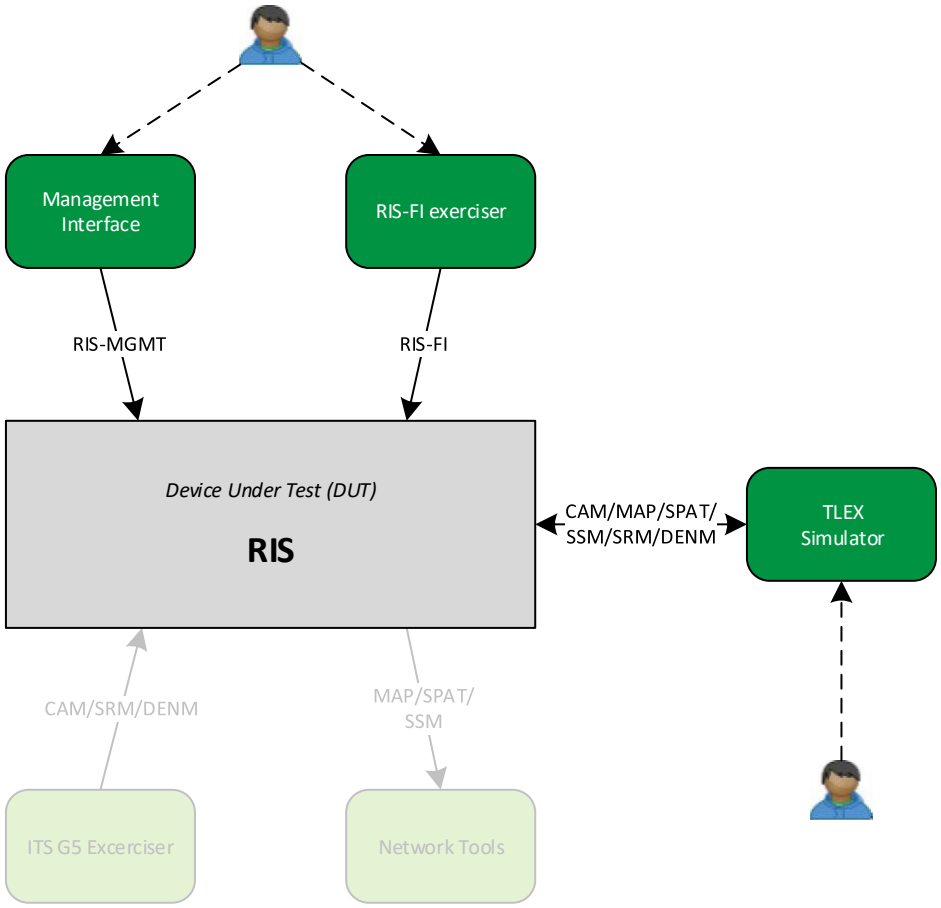

Figure 4: Connected test setup

6.7.2 Cooperative

When the cooperative method is used messages are exchanged via the NF-SAP. The ITS G5 exerciser is used to send C-ITS messages and Wireshark is used to monitor these messages.

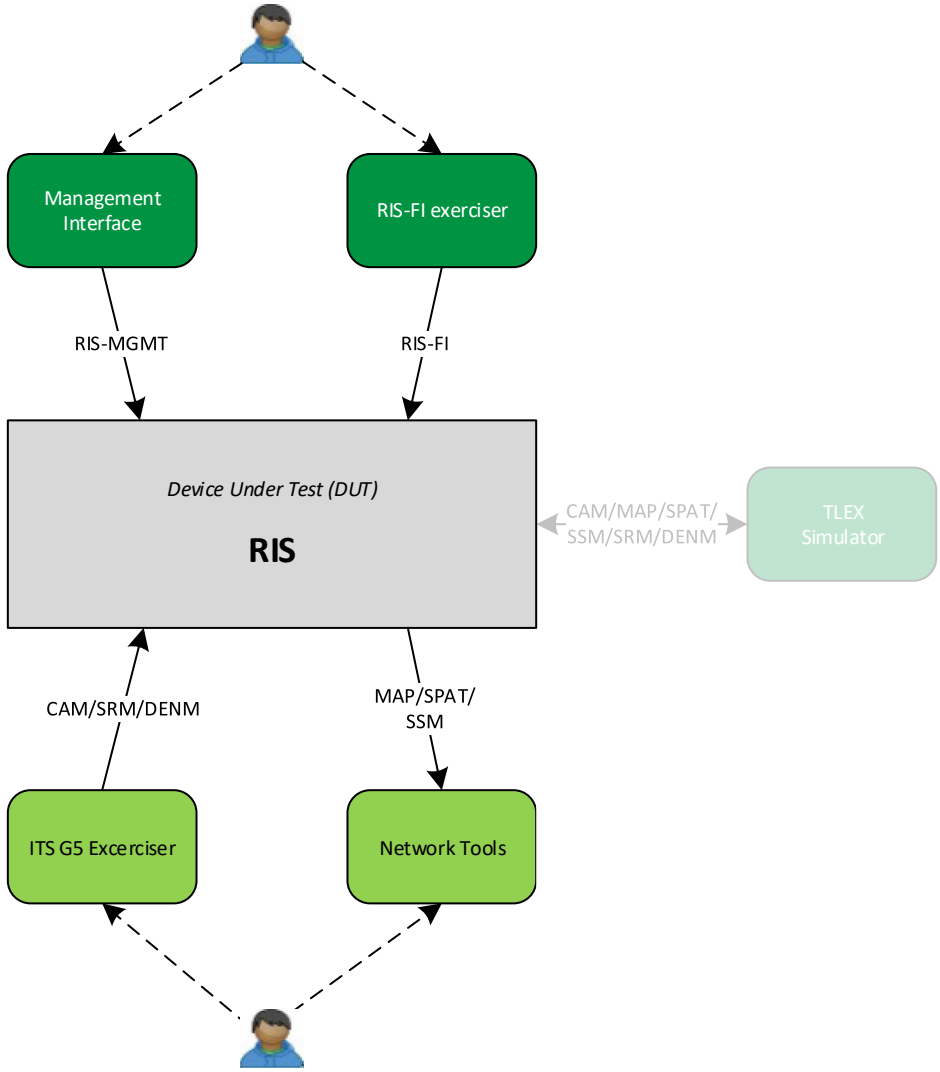

Figure 5: Cooperative test setup

6.7.3 Hybrid

When the hybrid method is used both TLEX and NF-SAP testing has to be done. For both interfaces all tests can be done.

For administration of the tests, create two instances of this test document to log the results.

7 Test scenarios

7.1 Registration

Fill in the following information about the DUT below, prior to executing the tests:

Vendor	
Product name	
Product code	
Hardware Versions	
Software Versions	
Provided documentation versions	
Test software versions	
Used Test hardware and setup	
RIS Compliance	[ ] Connected [ ] Hybrid Connected [ ] Cooperative [ ] Hybrid Cooperative

Table 7 : Device Under Test registration

7.2 RIS FI

7.2.1 Use Case: Monitoring of traffic

7.2.1.1 *Test Case: Start up and user configuration RIS*

Test case:	Start up and user configuration RIS		
ID:	RISFI.UC1.1		
Objective:	Verify the management interface is working and user management is in place		
Pre-conditions	The RIS is turned off, has no users configured and is connected to a network		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	**Verify** documentation for the RIS management interface (RIS-MGMT) is available		
2	Switch on the power of the RIS		
3	Connect to the management interface of the RIS using the administrator password. Create the user accounts given in Appendix A of the STP (📖 6) **User**: <User> **Password**: <Password> **Application type**: <Type>		
4	**Verify** that the accounts can be created**.**		
5	**Verify** in the audit log of the RIS that the administrator has logged in.		
Tested by:		**Date:**	

7.2.1.2 *Test Case: Intersection configuration*

Test case:	Intersection configuration		
ID:	RISFI.UC1.2		
Objective:	Verify the topology can be configured		
Pre-conditions	RIS is on and users are configured		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Using the management interface import the intersection topology from the external source and assure it contains the topology as described in § 5.3.1.		
2	**Verify** the complete topology file is accepted by the RIS and no errors occurred		
Tested by:		**Date:**	

7.2.1.3 *Test Case: Register ITS-CRA*

Test case:	Register ITS-CRA		
ID:	RISFI.UC1.3		
Objective:	Verify that the ITS-CRA can connect and register to the RIS-FI interface of the RIS, the connection is maintained and the access is logged in the security audit log of the RIS. Also checked is if the link is maintained.		
Pre-conditions	The RIS is started up and configured		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Configure the RIS-FI Exerciser to register using the **Consumer1** credentials.		
2	Use a secure TLS connection to connect and register to the RIS facilities **Note** the used port: _____		
3	**Verify** that the ITS-CRA is connected and registered to the RIS		
4	**Verify** protocol version conform the version listed in the referenced version of the RIS-FI IDD document 🕮 3 is returned.		
4	**Verify** that the connection is secured with TLS		
5	**Verify** that the audit log contains the start of the connection event and the result		
6	**Wait** at least one minute to verify that the keep alive messages function properly		
7	**Verify** that the application remains connected and registered		
8	**Verify** that the audit log contains no messages saying that the ITS-CRA is either deregistered, removed or that the connection is reestablished.		
Tested by:		**Date:**	

7.2.1.4 *Test Case: Subscription to ITS station events*

Test case:	Subscription to ITS station event		
ID:	RISFI.UC1.4		
Objective:	Verify two subscriptions in a single session can be made for ITS station events		
Pre-conditions	The RIS is started up and configured The ITS-CRA is registered to the RIS		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Subscribe to ITS station events by sending a **SubscribeObjects** method and subscribing to **ItsStation** of type **RISObjectType** with filter condition "**matches** != null"		
2	**Verify** an empty result is returned (because no stations exist in the LDM) with a subscription id **Note** the received **subscription** ID: _____ (ID1)		
3	**Verify** the response is received within 100 ms		
3	Subscribe to ITS station events by sending a **SubscribeObjects** method and subscribing to **ItsStation** of type **RISObjectType** with filter condition "**matches** != null"		
4	**Verify** an empty result is returned with a subscription id, different from the previous one **Note** the received **subscription** ID: _____ (ID2)		
5	**Verify** the response is received within 100 ms		
Tested by:		**Date:**	

7.2.1.5 *Test Case: Reception of ITS Station events*

Test case:	Reception of ITS Station events
ID:	RISFI.UC1.5
Objective:	Verify that ITS Station events are received (CAM), the data is validly mapped onto the intersection and that updated vehicle positions are received by the ITS applications. This is verified with 2 separate subscriptions in a single session. The generated vehicles drive in opposite direction and have to be spotted on several points in the lanes. Refer to *Figure 3* for the used references on the topology regarding positions.
Pre-conditions	The RIS is running The ITS-CRA is connect to the RIS and has two subscriptions to ItsStation

STEP	DESCRIPTION	PASS/FAIL	REMARKS/ACTIONS
1	Generate two vehicle CAM messages with the following specifications: Vehicle 1: • Positioned on lane 1 towards the signal group (SG) fc02, just after the start of the lane path • The **stationType** is **passengerCar (5)** • The **role** is **default (0)** • The **speed** value is 1250 (12,5 m/s) Vehicle 2: • Positioned on lane 8 towards the SG fc08, just after the start of the lane path • The **stationType** is **motorcycle (4)** • The **role** is **safetyCar (7)** • The **heading** is pointing towards the signal group • The **speed** value is 1194 (11,94 m/s)		
2	**Verify** that for each **subscription** ID two **NotifyObjects** are received within 500 ms		
3	**Verify** that the data shown in the **NotifyObjects** corresponds to the generated CAM messages. The ItsStation locationTime must be derived from generationDeltaTime in the CAM		
4	**Verify** that each **ItsStation** is mapped correctly on the intersection, e.g. **distance**, **lane**, **offset**, **signalGroup** etc.		

5	Update the position of Vehicle 2 so that the vehicle is on lane 3 and closes in on the stop line. Send a CAM message for this new position.		
6	**Verify** that for each **subscription** ID one **NotifyObjects** is received as the position of only one car is updated		
7	**Verify** that the data shown in the **NotifyObjects** corresponds to the generated CAM messages.		
8	**Verify** that the **ItsStation** is mapped correctly on the intersection, e.g. **distance**, **lane**, etc.		
9	**Verify** that the responses are received within 500 ms of sending the CAM messages		
Tested by:		**Date:**	

7.2.1.6 *Test Case: Unsubscribing from ITS Station events*

Test case:	Unsubscribing from ITS Station events		
ID:	RISFI.UC1.6		
Objective:	Verify that the subscription with ID2 can be removed without affecting any other subscriptions and the registration of the application		
Pre-conditions	The RIS is up and running The ITS-CRA is connect to the RIS with two subscriptions to ItsStation Two vehicles exist in the LDM		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Unsubscribe for the subscription with ID2 by sending an **UnsubscribeObjects** method with the **subscription** ID2.		
2	Update the position of Vehicle 2 so that the vehicle is still on lane 8 and is closer to the stop line. Do this by sending a CAM message		
3	**Verify** that for the subscription with ID1 a **NotifyObjects** is received and not for the subscription with ID2		
4	**Verify** that the correct data is shown in the **NotifyObjects** corresponding to the generated CAM messages.		
5	**Verify** that the **ItsStation** is mapped correctly on the intersection, e.g. **distance**, **lane**, **offset**, **signalGroup** etc.		
6	**Verify** that the response is received within 500 ms of sending the CAM messages		
Tested by:		**Date:**	

7.2.1.7 *Test Case: Deregister ITS-CRA*

Test case:	Deregister ITS-CRA		
ID:	RISFI.UC1.7		
Objective:	Verify that the ITS-CRA can deregister from the RIS		
Pre-conditions	The RIS is up and running The ITS-CRA is registered to the RIS		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Deregister **Consumer1** from the RIS		
2	**Verify** a deregistration reply is received		
3	**Verify** that the audit log contains a message saying that the ITS-CRA is deregistered		
Tested by:		**Date:**	

7.2.1.8 *Test Case: Map match verification*

Test case:	Map match verification LDM map
ID:	RISFI.UC1.8
Objective:	Verify vehicles are matched on the map
Pre-conditions	The RIS is up and running The ITS-CRA is registered to the RIS

STEP	DESCRIPTION	PASS/FAIL	REMARKS/ACTIONS
1	Subscribe to ITS station events by sending a **SubscribeObjects** method and subscribing to **ItsStation** of type **RISObjectType** with filter conditions "**matches** != null"		
2	**Verify** an empty result is returned with an subscription id		
3	Generate a vehicle CAM message with the following minimum specifications: Vehicle: • Positioned on lane 8 towards SG fc08. Add an offset so the vehicle is positioned towards lane 7 for about 30% • The **stationType** is **moped (3)** • The **role** is **default (0)** • The **heading** is pointing towards the signal group • The **speed** value is 1624 (16,24 m/s)		
4	**Verify** a **NotifyObjects** is received		
5	**Verify** that the correct data is shown in the **NotifyObjects** corresponding to the generated CAM message.		
6	Generate a vehicle CAM message with the following minimum specifications: Vehicle: • Positioned after the intersection on lane 10 in the extend of SG fc02 • The **stationType** is **lightTruck (7)** • The **role** is **agriculture (8)** • The **heading** is pointing away from the signal group • The **speed** value is 2222 (22,22 m/s)		

7	**Verify** a **NotifyObjects** is received		
8	**Verify** that the correct data is shown in the **NotifyObjects** corresponding to the generated CAM message.		
9	Generate a vehicle CAM message with the following minimum specifications: Vehicle: • Positioned in the middle of the conflict area • The **stationType** is **passengercar (5)** • The **role** is **default (0)** • The **heading** is south-east (152.3 degrees) • The **speed** value is 2322 (23,22 m/s) Generate a **bicycle** CAM message with the following minimum specifications: Bicycle: • Positioned between lane 1 and 10 in the conflict area • The **stationType** is **cyclist(2)** • The **role** is **default (0)** • The **heading** is west (270.0 degrees) • The **speed** value is 1222 (12,22 m/s)		
10	**Verify** **NotifyObjects** is received		
11	**Verify** that the correct data is shown in the **NotifyObjects** corresponding to the generated CAM message. Among the matched lane(s) for both CAMs also a match must be present on Lane 0.		
Tested by:		**Date:**	

7.2.1.9 *Test Case exception: Invalid subscription parameters*

Test case:	Invalid subscription parameters		
ID:	RISFI.UC1.9		
Objective:	Verify an error is returned when an invalid subscription parameter is send		
Pre-conditions	The RIS is up and running The ITS-CRA is registered to the RIS		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Subscribe to ITS station events by sending a **SubscribeObjects** method and subscribing to the non-existing type 8 of **RISObjectType**.		
2	**Verify** an error is returned with code set to **InvalidAttributeValue** of 8		
Tested by:		**Date:**	

7.2.2 Use Case: Bus priority handling based on CAM

7.2.2.1 *Test Case: Receive CAM from bus ITS station*

Test case:	Receive CAM from bus ITS station
ID:	RISFI.UC2.1
Objective:	The RIS receives information about busses in the neighborhood via Cooperative Awareness Messages (CAM). An ITS-A (ITS Application) can, based upon this information, request for priority at the TLC facilities to give way to these busses.
Pre-conditions	The RIS is up and running The ITS-CRA is registered to the RIS

STEP	DESCRIPTION	PASS/FAIL	REMARKS/ACTIONS
1	Generate a vehicle CAM message with the following specifications: Vehicle 1: • Positioned on lane 2 towards the SG fc03, just after the start of the lane path • The **stationType** is **passengerCar (5)** • The **role** is **default (0)** • The **heading** is pointing towards the signal group • The **speed** value is 1250 (12,5 m/s) Vehicle 2: • Positioned on lane 8 towards the SG fc08 just after the start of the lane path • The **stationType** is **bus (6)** • The **role** is **publicTransport (1)** • The **heading** is pointing towards the signal group • The **speed** value is 1310 (13,1 m/s) • The **PublicTransport lineNr** is 5 Vehicle 3: • Positioned on lane 7 towards the SG fc07 just after the start of the lane path • The **stationType** is **bus (6)** • The **role** is **publicTransport (1)** • The **heading** is pointing towards the signal group • The **speed** value is 1020 (10,2 m/s) • The **PublicTransport lineNr** is 12 Vehicle 4:		

	• Positioned before the start of lane 5, in the direction of SG fc05 (so It is not yet on the local map!) • The **stationType** is **bus (6)** • The **role** is **publicTransport (1)** • The **heading** is pointing towards the signal group • The **speed** value is 1150 (11,5 m/s) • The **PublicTransport lineNr** is 33		
2	Subscribe to ITS station events by sending a **SubscribeObjects** method and subscribing to **ItsStation** of type **RISObjectType** with filter conditions: • "**matches** != null" and • **stationType** == 6 (only busses)		
3	**Verify** a result is returned with a **subscription** id and with only the data of vehicle 2 and vehicle 3.		
4	**Verify** that only **ItsStation** of vehicle2 and 3 are mapped correctly on the intersection, by checking the **distance**, **lane, and offset**		
5	**Verify** that the CAM messages are logged		
6	Resend the CAM from Vehicle 3 using the exact same parameters		
7	**Verify** that no update **NotifyObjects** is received		
8	Update the position of Vehicle 4 so that the vehicle is on lane path 5 and closes in on the stop line. Do this by sending a CAM message.		
9	**Verify** that the CAM messages are logged		
10	**Verify** that a **NotifyObjects** is received for the **subscription** with the data of vehicle 4		
11	**Verify** that the correct data is shown in the **NotifyObjects** corresponding to the generated CAM messages.		
12	**Verify** that the **ItsStation** is mapped correctly on the intersection, e.g. **distance**, **lane, offset**, etc.		
13	**Verify** that the responses are received within 500 ms of sending the CAM messages		
Tested by:		**Date:**	

7.2.2.2 *Test Case: Recovery after power-cycle*

Test case:	Recovery after power-cycle		
ID:	RISFI.UC2.2		
Objective:	Verify recovery of the RIS-FI interface after a RIS power-cycle with operational RIS-FI connections.		
Pre-conditions	The RIS is up and running The ITS-CRA is registered to the RIS No vehicles are present in the RIS facilities		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Subscribe to ITS station events by sending a **SubscribeObjects** method and subscribing to **RISObjectType** of type **ItsStation**		
2	**Verify** an empty result is returned with a **subscription** id		
3	Send a vehicle CAM message with the following specifications: • Positioned on lane 8 towards the SG8, just after the start of the lane path • The **stationType** is **bus (6)** • The **role** is **publicTransport (1)** • The **heading** is pointing towards the signal group • Enter a **speed** value • The **PublicTransport lineNr** is 05		
4	**Verify** that a **NotifyObjects** is received for the **subscription** with the data of the generated vehicle		
5	Power-cycle the RIS and wait for the RIS to recover		
6	Repeat step 1 till 3		
7	**Verify** that a **NotifyObjects** is received for the **subscription** with the data of the generated vehicle		
Tested by:		**Date:**	

7.2.3 Use Case: Bus priority handling based on SRM

7.2.3.1 *Test case: public transport SRM and SSM handling*

Test case:	Public transport SRM and SSM handling
ID:	RISFI.UC3.1
Objective:	The RIS receives information about busses in the neighborhood via Cooperative Awareness Messages (CAM) and Signal Request Messages (SRM). An ITS-A can, based upon this information, request for priority at the TLC facilities to give way to these busses
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS A bus is approaching the intersection broadcasting CAM and SRM.

STEP	DESCRIPTION	PASS/FAIL	REMARKS/ACTIONS
1	Subscribe to **PrioritizationRequest** events by sending a **SubscribeObjects** method with type **PrioritizationRequest** with filter condition "**role** == **publicTransport**"		
2	**Verify** a result is returned containing only a subscription id		
3	Subscribe to **ItsStation** events by sending a **SubscribeObjects** method with type **ItsStation** with filter condition "**matches** != null"		
4	**Verify** a result is returned containing only a subscription id		
5	As long as the test is running every 1.5 seconds generate a vehicle CAM message with the following specifications: • Positioned on lane 8 towards the SG fc08 just after the start of the lane path • The **StationID** is 7007 • The **StationType** is **bus (6)** • The **VehicleRole** is **publicTransport (1)** • **exteriorLights** is "00010000" (rightTurnSignalOn (3)) • The **Heading** is pointing towards the signal group • The **speed** value is 1400 (14 m/s) • The **PublicTransport** ▪ **embarkationStatus** is false ▪ **ptActivation.PtActivationType** is 3 ▪ **ptActivation.PtActivationData.lineNr** is 5 ▪ rest of **ptActivation.PtActivationData** is 0		

	Generate a single SRM message containing the following specification:
	• **sequenceNumber** is 5
	• **request**
	  ○ **RequestID** is 104
	  ○ **requestType** is priorityRequest (1)
	  ○ **inBoundLane** is connection 6
	• **requestor**
	  ○ **id.stationID** is 7007
	  ○ **type.role** is publicTransport(1)
	  ○ **type.subrole** is requestSubRole1 (1), -- bus
	  ○ **type.request** is requestImportanceLevel2(2)
	  ○ **routeName** is “T101”
	  ○ **transitStatus** is doorOpen 00010000 (3)
	  ○ **transitSchedule** is -60
	• **minute** corresponds to current time + 10s
	• **second** corresponds to current time + 10s
	• **duration**: This field must be omitted		

6	**Verify** that a **NotifyObjects** is received for the **subscription** for a **ItsStation** object		
7	**Verify** that the data shown in the **NotifyObjects** corresponds to the generated CAM message**.** (Beware of the **turn** field which must be right(3).		
8	**Verify** that a **NotifyObjects** is received for the **subscription** for a **PrioritizationRequest** object.		
9	**Verify** that the correct data is shown in the **PrioritizationRequest** object for fc08 corresponding to the CAM and the SRM message. Among the other fields, especialy verify the <OPT> fields: **signalGroup** is TINT1_fc08, **routeName** is “T101”, **TransitStatus**.**loading** is false, **TransitStatus**.**anANDAuse** is false, **TransitStatus**.**aBikeLoad** is false, **TransitStatus**.**doorOpen** is true, **TransitStatus**.**charging** is false,		

	TransitStatus.atStopLine is false, **punctuality** is -60, **importance** is 2.		
10	**Verify** that the responses are received within 500 ms of sending the CAM and SRM messages		
11	Send an **UpdateObjects** method with an update for the **ActivePrioritization** object with the following minimum specifications: ▪ **Id** is (7007_104) ▪ The **prioritizations** as follow ▪ **sequenceNumber is 5** ▪ **priostate** is granted (4)		
12	**Verify** an empty **updateObjects** result is returned		
13	**Verify** an SSM message is broadcasted		
14	**Verify** the SSM message contains the data configured at step 5 (see 12 Appendix E)		
15	**Verify** a notification has been received for the **itsStation** subscription that contains expired[] set to id 7007		
16	**Verify** a notification has been received for the **prioritizationRequest** subscription that contains expired[] set to id 7007_104, 73s after the SRM is sent		
17	Send a **RequestObjects** method with **filter** for type **ActivePrioritization**		
18	**Verify** an empty result is returned from the **RequestObjects**		
Tested by:		**Date:**	

7.2.3.2 *Test case: emergency vehicle SRM and SSM handling*

Test case:	Emergency vehicle SRM and SSM handling
ID:	RISFI.UC3.2
Objective:	The RIS receives information about an emergency vehicle in the neighborhood via Cooperative Awareness Messages (CAM) and Signal Request Messages (SRM). An ITS-A can, based upon this information, request for priority at the TLC facilities to give way to these vehicles
Pre-conditions	The RIS is up and running

	The ITS-PRA is registered to the RIS

STEP	DESCRIPTION	PASS/FAIL	REMARKS/ACTIONS
1	Subscribe to **PrioritizationRequest** events by sending a **SubscribeObjects** method with type **PrioritizationRequest** with filter condition "**role** == **emergency**"		
2	**Verify** a result is returned containing only a subscription id		
3	Subscribe to **ItsStation** events by sending a **SubscribeObjects** method with type **ItsStation** with filter condition "**matches** != null"		
4	**Verify** a result is returned containing only a subscription id		
5	Generate a vehicle CAM message with the following specifications: • Positioned on lane 1 towards the SG fc02 just after the start of the lane path • The **StationID** is 9008 • The **StationType** is **passengerCar (5)** • The **VehicleRole** is **emergency (6)** • The **Heading** is pointing towards the signal group • The **speed** value is 1800 (18 m/s) • The **Emergency** ▪ **lightBarSirenInUse is "11"** Generate an SRM message containing the following specification: • **sequenceNumber** is **12** • **request** ○ **RequestID** is 90 ○ **requestType** is priorityRequest (1) ○ **inBoundLane** is approach 1 • **requestor** ○ **id.stationID** is 9008 ○ **type.role** is emergency(6) ○ **type.subrole** is requestSubRole5 (5), -- blue light and siren • **minute** corresponds to current time + 20s • **second** corresponds to current time + 20s • **duration** is 0		

6	**Verify** that a **NotifyObjects** is received for the **subscription** for a **ItsStation** object		
7	**Verify** that the data shown in the **NotifyObjects** corresponds to the generated CAM message**.** (Beware of the **turn** field which must be straight (2) and the **roleAttributes.lightBarActivated** is true, **roleAttributes.sirenActivated** is true.)		
8	**Verify** that a **NotifyObjects** is received for the **subscription** for 1 **PrioritizationRequest** object.		
9	**Verify** that the correct data is shown in one **PrioritizationRequest** object corresponding to the CAM and the SRM message. Especially check the fields: **approach** is 1 **signalGroup** is not filled in		
10	**Verify** that the responses are received within 500 ms of sending the CAM and SRM messages		
11	Send an **UpdateObjects** method with an update for the **ActivePrioritization** object with the following minimum specifications: ▪ **Id** is (9008_90) ▪ The **prioritizations** as follow ▪ **sequenceNumber is 12** ▪ **priostate** is granted (4)		
12	**Verify** an empty **updateObjects** result is returned		
13	**Verify** an SSM message is broadcasted		
14	**Verify** the SSM message contains the data corresponding to the SRM and the granted state.		
15	Before the previous CAM and SRM expire send: the same CAM message with updated position: • Positioned on lane 1 towards the SG fc02 about in the middle between start and the stopline of the approach path Generate the same SRM message only with update request : • **sequenceNumber** is **13**		

	• **request** ○ **requestType** is priorityRequestUpdate (2) • **minute** corresponds to current time + 5s • **second** corresponds to current time + 5s		
16	**Verify** that an updated **NotifyObjects** is received for the **subscription** for a **ItsStation** object		
17	**Verify** that an updated **NotifyObjects** is received for the **subscription** for a single **PrioritizationRequest** object with **PriorityRequestType is** update(2)		
18	Wait until the itsStation notification is received with **expired[]** containing ID 9008		
19	**Verify** a notification has been received for the **prioritizationRequest** subscription that contains expired[] set to id 9008_90, 5s after the second SRM is sent		
20	Send a **RequestObjects** method with **filter** for type **ActivePrioritization**		
21	**Verify** an empty result is returned from the **RequestObjects**		
Tested by:		**Date:**	

7.2.3.3 *Test Case Exceptions: Illegal update of ActivePrioritization*

Test case:	Unauthorized and illegal update of **ActivePrioritization**		
ID:	RISFI.UC3.3		
Objective:	Verify that an unauthorized user cannot update the **SignalGroup** and verify illegal values are not accepted.		
Pre-conditions	The RIS is up and running The ITS-CRA is registered to the RIS		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Generate a vehicle CAM message with the following specifications: • Positioned on lane 1 towards the SG fc02 just after the start of the lane path • The **StationID** is 9009 • The **StationType** is **passengerCar (5)** • The **VehicleRole** is **emergency (6)** • The **Heading** is pointing towards the signal group • The **speed** value is 1800 (18 m/s) • The **Emergency**		

	▪ **lightBarSirenInUse is "11"**
	
	Generate an SRM message containing the following specification:
	
	• **sequenceNumber** is 14
	• **request**
	  ○ **RequestID** is 91
	  ○ **requestType is** priorityRequest (1)
	  ○ **inBoundLane** is approach 1
	• **requestor**
	  ○ **id.stationID** is 9009
	  ○ **type.role** is emergency(6)
	• **minute** corresponds to current time + 60s
	• **second** corresponds to current time + 60s
	• **duration** is 0		

2	Send an **UpdateObjects** method with an update for the **ActivePrioritization** object with the following minimum specifications:
	
	▪ **Id** is (9009_91)
	  ▪ The **prioritizations** as follow
	  ▪ **sequenceNumber is 14**
	  ▪ **priostate** is granted (4)		

3	**Verify** a result is returned in which the error code is set to **ProtocolErrorCode** of 2006 or 1 (Not authorized for this object) and that the application is disconnected		
4	**Verify** no SSM is transmitted.		
5	Register as ITS-PRA		
6	Send an **UpdateObjects** method with an update for the **ActivePrioritization** object with the following minimum specifications: ▪ **Id** is (9009_91)		

	▪ The **prioritizations** as follow ▪ **sequenceNumber is 128** ▪ **priostate** is granted (4)		
7	**Verify** error code InvalidAttributeValue(8) or Parameter out of range (2005) is returned and that the application is disconnected.		
8	**Verify** no SSM is transmitted.		
9	Send an **UpdateObjects** method with an update for the **ActivePrioritization** object with the following minimum specifications: ▪ **Id** is (9009_91) ▪ The **prioritizations** as follow ▪ **sequenceNumber is 14** ▪ **prioState is 8**		
10	**Verify** error code InvalidAttributeValue(8) or Parameter out of range (2005) is returned and that the application is disconnected		
11	**Verify** no SSM is transmitted.		
12	Send an **UpdateObjects** method with an update for the **ActivePrioritization** object with the following minimum specifications: ▪ **Id** is (9009_91) ▪ The **prioritizations** as follow ▪ **sequenceNumber is 14.0** ▪ **prioState is granted**		
13	**Verify** error code InvalidAttributeType (7) or Parameter out of range (2005) is returned and that the application is disconnected		
14	**Verify** no SSM is transmitted.		
15	Send an **UpdateObjects** method with an update for the **ActivePrioritization** object with the following minimum specifications: ▪ **Id** is (9009_91) ▪ The **prioritizations** as follow ▪ **sequenceNumber is 14**		

	▪ **prioState is false**		
16	**Verify** error code InvalidAttributeType (7) or Parameter out of range (2005) is returned and that the application is disconnected		
17	**Verify** no SSM is transmitted.		
Tested by:		**Date:**	

7.2.3.4 *Test Case Exceptions: Illegal update of PrioritizationRequest*

Test case:	Unauthorized and illegal update of **PrioritizationRequest**		
ID:	RISFI.UC3.4		
Objective:	Verify that a PrioritizationRequest cannot be updated.		
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Generate a vehicle CAM message with the following specifications: • Positioned on lane 1 towards SG02 just after the start of the lane path • The **StationID** is 9010 • The **StationType** is **passengerCar (5)** • The **VehicleRole** is **emergency (6)** • The **Heading** is pointing towards the signal group • The **speed** value is 1800 (18 m/s) • The **Emergency** ▪ **lightBarSirenInUse is "11"** Generate an SRM message containing the following specification: • **sequenceNumber is 15** • **request** ○ **RequestID** is 92 ○ **requestType** is priorityRequest (1)		

	○ **inBoundLane** is approach 1 • **requestor** ○ **id.stationID** is 9010 ○ **type.role** is emergency(6) • **minute** corresponds to current time + 60s • **second** corresponds to current time + 60s • **duration** is 0		
2	Send an **UpdateObjects** method with an update for the **PrioritizationRequest** object with the following minimum specifications: • **Id** is 9010_92 • **sequenceNumber** is 15 • **requestType** is cancellation(3) • **itsStation** is 9010 • **intersection** is 1 • **role** is emergency(6) • **subrole** is unknown(0) • **eta** is corresponding to current time		
3	**Verify** a result is returned in which the error code is set to **ProtocolErrorCode** of 2006 or 1 (Not authorized for this object) and that the application is disconnected.		
4	Reconnect and register again		
5	Send a **RequestObjects** method with **filter** for type **PrioritizationRequest**		
6	Verify a **PrioritizationRequest** is returned with data corresponding to the SRM in step 1		
Tested by:		**Date:**	

7.2.4 Use Case: Create an ItsEvent

7.2.4.1 *Test Case: Create a single ItsEvent*

Test case:	Create a single ItsEvent		
ID:	RISFI.UC4.1		
Objective:	Verify that DENM messages are sent once when no repetition interval is set		
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Create a **CreateEvents** with an **ItsEvent** with the following specification: • The **eventType** is **adverseWeatherCondition-ExtremeWeatherCondition (17)** • The **eventSubType** is 0 • The **validityDuration** is 15 • The **relevanceDistance** is 400.0 • The **trafficDirection** is **upstreamTraffic (1)** • **For eventPosition is** use **E1** described in *Table 2: Events reference* • For **traces** use trace **T1** described in *Table 2: Events reference* • The **detectionTime** is current time Send the **CreateEvents** method to the RIS		
2	**Verify** a result is returned with a subscription ID for the defined event		
3	**Verify** a DENM message has been broadcasted once		
4	**Verify** that the content of the DENM message corresponds to the data in the **createEvents** object		
Tested by:		**Date:**	

7.2.4.2 *Test Case: Create multiple ItsEvents*

Test case:	Create multiple ItsEvents
ID:	RISFI.UC4.2
Objective:	Events are used to inform ITS stations about potentially dangerous situations (e.g. Traffic jam ahead, animal on the road, bad weather condition etc.). In the case that an ITS application detects such a dangerous situation, it can create an ItsEvent object in the RIS. Verify DENM messages are transmitted with correct content based on a RIS-FI **CreateEvents** message with two Events. Assure the events are removed after their lifetime expired
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS

STEP	DESCRIPTION	PASS/FAIL	REMARKS/ACTIONS
1	Send a **CreateEvents** with two **ItsEvents** with the following specification: Event 1: • The **eventType** is **unknown (0)** • The **eventSubType** is 0 • The **validityDuration** is 15 • The **relevanceDistance** is 400.0 • The **trafficDirection** is **upstreamTraffic (1)** • The **repetitionInterval** is 3000 Event 2: • The **eventType** is **roadworks (3)** • The **eventSubType** is 1 • The **validityDuration** is 10 • The **relevanceDistance** is 600.0 • The **trafficDirection** is **upstreamTraffic (1)** • The **repetitionInterval** is 1000 For both events the following applies: • For **eventPosition** use E1 described in *Table 2: Events reference*, enter the data in WGS84 coordinates • For **traces** use T1 described in *Table 2: Events reference*		

	• For **destinationArea** is: ▪ **centre** is same as **eventPosition** ▪ **majorAxis** is 500.0 ▪ **minorAxis** is 500.0 ▪ **angle** is 0.0 ▪ **circular** is true • The **detectionTime** for both are set to the current time Send the **CreateEvents** method to the RIS		
2	**Verify** a result is returned with a subscription ID for each defined event ID 1: _____ ID 2: _____		
3	**Verify** the response is received within 100 ms		
4	**Verify** that DENM messages are broadcasted with the configured **repetitionInterval** (beware, two events are configured)		
5	**Verify** that after **validityDuration** no DENM are transmitted for the specific events		
6	**Verify** that the content of the DENM messages corresponds to the data in the **createEvents** object **Verify** that the **eventPostition** in the DENM is represented as WGS84 coordinates		
7	**Verify** that the **destinationArea** is correctly translated to LDM properties		
8	Assure the **validityDuration** for each event has expired Send a **RequestObjects** method for type **ItsEvent** (2)		
9	**Verify** an empty response is returned as no events exist		
10	**Verify** that the transmitted DENM messages are logged		
Tested by:		**Date:**	

7.2.4.3 *Test Case: Handling of unknown attribute*

Test case:	Handling of unknown attribute
ID:	RISFI.UC4.3
Objective:	Verify that the RIS Facilities can handle unknown attributes
Pre-conditions	The RIS is up and running

	The ITS-CRA is registered to the RIS		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Create a **CreateEvents** with an **ItsEvent** with the following specification: Event 1: • The **eventType** is **dangerousEndOfQueue (27)** • The **eventSubType** is 0 • The **validityDuration** is 15 • The **relevanceDistance** is 400.0 • The **trafficDirection** is **oppositeTraffic (3)** • **For eventPosition** use **E1** described in *Table 2: Events reference* • For **traces** use trace **T1** described in *Table 2: Events reference* • The **detectionTime** is current time • The ***scoopAttribute*** is none Send the **CreateEvents** method to the RIS		
2	**Verify** that the RIS Facilities ignores the attribute ***scoopAttribute*** and continues to process the remaining attributes		
3	**Verify** a DENM message has been broadcasted once		
4	**Verify** that the content of the DENM message corresponds to the data in the **createEvents** object		
Tested by:		**Date:**	

7.2.4.4 *Test Case: Handling of missing mandatory attribute*

Test case:	Handling of missing mandatory attribute		
ID:	RISFI.UC4.4		
Objective:	Verify that the RIS-FI can handle missing mandatory attributes		
Pre-conditions	The RIS is up and running The ITS-CRA is registered to the RIS		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Create a **CreateEvents** with an **ItsEvent** with the following specification:		

	Event 1: • The **eventType** is **adverseWeatherCondition-Visibility (18)** • The **eventSubType** is 0 • The **validityDuration** is 15 • The **trafficDirection** is **allTrafficDirections (0)** • **For eventPosition** use **E1** and **E2** described in *Table 2: Events reference* • For **traces** use trace **T1** described in *Table 2: Events reference* • The **detectionTime** is current time Send the **CreateEvents** method to the RIS		
2	**Verify** that the RIS facilities sends in response a message with the error code 6 (*MissingAttribute)* as the **relevanceDistance** is missing		
3	**Verify** no DENM message has been broadcasted		
Tested by:		**Date:**	

7.2.4.5 *Test Case Exception: Unauthorized creation of event*

Test case:	Unauthorized creation of event		
ID:	RISFI.UC4.5		
Objective:	Verify an event cannot be created by an ITS-CRA application		
Pre-conditions	The RIS is up and running The ITS-CRA is registered to the RIS		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Create a **CreateEvents** with an **ItsEvent** with the following minimum specification: Event 1: • The **eventType** is **slowVehicle (26)** • The **eventSubType** is 0 • The **validityDuration** is 18 • The **relevanceDistance** is 330.0 • The **trafficDirection** is **oppositeTraffic (3)** • **For eventPosition is** use **E1** described in *Table 2: Events reference* • For **traces** use trace **T1** described in *Table 2: Events reference* • The **detectionTime** is current time Send the **CreateEvents** method to the RIS		
2	**Verify** a result is returned with error code 2006 or 1(Not authorized)		
3	**Verify** no DENM message have been broadcasted		
4	Deregister from the RIS		
Tested by:		**Date:**	

7.2.5 Use Case: Update an ItsEvent

7.2.5.1 *Test Case: Update two events*

Test case:	Update two events
ID:	RISFI.UC5.1
Objective:	Verify two DENM messages are updated based on an **UpdateObjects** message with updates for all fields or for a single field for 2 events
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS

STEP	DESCRIPTION	PASS/FAIL	REMARKS/ACTIONS
1	Create a **CreateEvents** with two **ItsEvents** with the following specification: Event 1: • The **eventType** is **trafficCondition (1)** • The **eventSubType** is 0 • The **validityDuration** is 15 • The **relevanceDistance** is 400.0 • The **trafficDirection** is **upstreamTraffic (1)** • The **repetitionInterval** is 3000 • For **eventPosition** use E1 described in *Table 2: Events reference* • For **traces** use T1 described in *Table 2: Events reference* Event 2: • The **eventType** is **humanPresenceOnTheRoad (12)** • The **eventSubType** is 1 • The **validityDuration** is 10 • The **relevanceDistance** is 600.0 • The **trafficDirection** is **upstreamTraffic (1)** • The **repetitionInterval** is 1000 • For **eventPosition** use E2 described in *Table 2: Events reference* • For **traces** use T3 described in *Table 2: Events reference*		

	For both events the following applies: • The **Timestamp** for both are set to the current time Send the **CreateEvents** method to the RIS		
2	**Verify** a result is returned with a subscription ID for each defined event ID 1: _____ ID 2: _____		
3	**Verify** that DENM messages are broadcasted		
4	**Verify** that the content of the DENM messages corresponds to the data in the **createEvents** object		
5	Send after 5 seconds an **UpdateObjects** with the following specifications: Event 1 update (ID 1): • The **eventType** is **wrongWayDriving (14)** • The **eventSubType** is 2 • The **validityDuration** is 30 • The **relevanceDistance** is 1010.0 • The **trafficDirection** is **downstreamTraffic (2)** • The **repetitionInterval** is 1500 • For **eventPosition** use E1 described in *Table 2: Events reference* • For **traces** use T2 described in *Table 2: Events reference* Event 2 update (ID 2): • The **eventType** is **dangerousSituation (99)** • The **eventSubType** is 3 • The **validityDuration** is 28 • The **relevanceDistance** is 1500.0 • The **trafficDirection** is **allTrafficDirections (0)** • The **repetitionInterval** is 1200 • For **eventPosition** use E3 described in *Table 2: Events reference* • For **traces** use T4 described in *Table 2: Events reference* For both events the following applies:		

	• The **detectionTime** for both are set to the current time		
6	**Verify** an empty **updateObjects** result is returned		
7	**Verify** that DENM messages are broadcasted with the updated **repetitionInterval**		
8	**Verify** the DENM contents are changed to the updated values in the **ObjectStateUpdate**		
9	Send after 5 seconds an **UpdateObjects** with the following specifications: Event 1 update (ID 1): • The **repetitionInterval** is 500 Event 2 update (ID 2): • The **repetitionInterval** is 1000		
10	**Verify** an empty **updateObjects** result is returned		
11	**Verify** that DENM messages are broadcasted with the updated **repetitionInterval**		
12	**Verify** that after the **validityDuration** (from step 9) no DENM are transmitted for the specific events		
Tested by:		**Date:**	

7.2.5.2 *Test Case: Update single field*

Test case:	Event update 1		
ID:	RISFI.UC5.2		
Objective:	Verify that with UpdateObjects one field of an object can be updated and that this field is also updated in the DENM		
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Create a **CreateEvents** with an **ItsEvent** with the following minimum specification:		

	• The **eventType** is **adverseWeatherCondition-Precipitation (19)** • The **eventSubType** is 0 • The **validityDuration** is 15 • The **relevanceDistance** is 400.0 • The **trafficDirection** is **upstreamTraffic** • **For eventPosition is** use **E1** described in *Table 2: Events reference* • For **traces** use trace **T1** described in *Table 2: Events reference* • The **detectionTime** is current time Send the **CreateEvents** method to the RIS		
2	**Verify** a result is returned with a subscription ID for the defined event ID 1: _____		
3	Send after 5 seconds an **UpdateObjects** to change the **eventType** to **humanProblem (93)**		
4	**Verify** an empty **updateObjects** result is returned **Verify** that the **eventType** is updated to **humanProblem** in the DENM and that the unchanged fields contain their original value		
5	Deregister from the RIS		
Tested by:		**Date:**	

7.2.5.3 *Test Case Exception: ItsEvent object ownership*

Test case:	ItsEvent object ownership		
ID:	RISFI.UC5.3		
Objective:	Verify that an event cannot be changed by another user of the RIS-FI An event is created by user1 and user2 tries to update or terminate the event		
Pre-conditions	The RIS is up and running The ITS-PRA1 is registered to the RIS as **Provider2** The ITS-PRA2 is registered to the RIS as **Provider3**		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
6	Send a **CreateEvents** with an **ItsEvent** with the following specification:		

	Event 1: • The **eventType** is **signalViolation (98)** • The **eventSubType** is 0 • The **validityDuration** is 15 • The **relevanceDistance** is 400.0 • The **trafficDirection** is **upstreamTraffic** • **For eventPosition is** use **E1** described in *Table 2: Events reference* • For **traces** use trace **T1** described in *Table 2: Events reference* • The **detectionTime** is current time Send the **CreateEvents** object to the RIS		
7	**Verify** a result is returned with a subscription ID for the defined event ID 1: _____		
8	Send an **UpdateObjects** object from ITS-PRA2 with an update changing the **eventType** to **slowVehicle** for the event created in step 1		
9	**Verify** ITS-PRA2 receives a result with the error code set to **ProtocolErrorCode** of 2006 or 1 (NotAuthorized) **Verify** the **eventType** is not updated in the transmitted DENM		
10	Send with ITS-PRA2 a **TerminateEvent** for the event created in step 1		
11	**Verify** a result is returned in which the error code is set to **ProtocolErrorCode** 2006 or 1 (NotAuthorized) **Verify** the event is still broadcasted through DENM messages		
12	Deregister **Provider2** from the RIS		
13	Deregister **Provider3** from the RIS		
Tested by:		**Date:**	

7.2.5.4 *Test Case Exception: ItsEvent check validity of fields*

Test case:	ItsEvent check validity of fields
ID:	RISFI.UC5.4
Objective:	Verify that the RIS checks for mandatory fields missing, field types, field values and incorrect object references

Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Send a **CreateEvents** with an **ItsEvent** with the following specification: • The **eventType** is **stationaryVehicle (94)** • The **eventSubType** is 0 • The **validityDuration** is 900 • The **relevanceDistance** is 800.0 • The **trafficDirection** is **upstreamTraffic** • **For eventPosition** use **E1** described in *Table 2: Events reference* • For **traces** use trace **T1** described in *Table 2: Events reference* • The **detectionTime** is current time		
2	**Verify** a result is returned with a subscription ID for the defined event ID 1: _____		
3	Resend the **CreateEvents** from step 1 but omit **validityDuration**		
4	**Verify** a result is returned in which the error code is set to **ProtocolErrorCode** 6 (MissingAttribute) (**validityDuration** is missing)		
5	**Repeat** step 3 and 4, but omit **eventType**		
6	**Repeat** step 3 and 4, but omit **eventSubType**		
7	**Repeat** step 3 and 4, but omit **trafficDirection**		
8	**Repeat** step 3 and 4, but omit **relevanceDistance**		
9	**Repeat** step 3 and 4, but omit **eventPosition**		
10	**Repeat** step 3 and 4, but omit **detectTime**		
11	Resend the **CreateEvents** from step 1 but set the **validityDuration** to 86401		
12	**Verify** a result is returned in which the error code is set to **ProtocolErrorCode** of 8 (**InvalidAttributeValue**)(**validityDuration** is out of range)		
13	**Repeat** step 11 and 12, but set the **eventType** to 20		
14	**Repeat** step 11 and 12, but set the **eventType** to 28		
15	**Repeat** step 11 and 12, but set the **eventType** to 90		
16	**Repeat** step 11 and 12, but set the **eventSubType** to 5		
17	**Repeat** step 11 and 12, but set the **trafficDirection** to 4		
18	**Repeat** step 11 and 12, but set the **relevanceDistance** to 429496729.6		

19	**Repeat** step 11 and 12, but set the **eventPosition** to { -90.000001, 0.000000, 0.000)		
20	**Repeat** step 11 and 12, but set the **eventPosition** to { 90.000001, 0.000000, 0.000)		
21	**Repeat** step 11 and 12, but set the **eventPosition** to { 0.000000, -180.000001, 0.000)		
22	**Repeat** step 11 and 12, but set the **eventPosition** to { 0.000000, 180.000001, 0.000)		
23	**Repeat** step 11 and 12, but set the **eventPosition** to { 0.000000, 0.000000, -100.001)		
24	**Repeat** step 11 and 12, but set the **eventPosition** to { 0.000000, 0.000000, -8000.001)		
25	**Repeat** step 11 and 12, but set the **detectionTime** to 18446744073709551616		
26	Resend the **CreateEvents** from step 1 but set **validityDuration** to **"twohundred"**		
27	**Verify** a result is returned in which the error code is set to **ProtocolErrorCode** 7 (**InvalidAttributeType**)		
28	**Repeat** step 26 and 27, but set the **eventType** to 0.1		
29	**Repeat** step 26 and 27, but set the **eventType** to "1"		
30	**Repeat** step 26 and 27, but set the **eventSubType** to "5"		
31	**Repeat** step 26 and 27, but set the **trafficDirection** to true		
32	**Repeat** step 26 and 27, but set the **relevanceDistance** to "534		
33	**Repeat** step 26 and 27, but set the **eventPosition** to { "0.000001", 0.000000, 0.000)		
34	**Repeat** step 26 and 27, but set the **eventPosition** to { 80.00000, "0.000400", 0.000)		
35	**Repeat** step 26 and 27, but set the **eventPosition** to { 0.000000, 0.000001, "4.000")		
36	**Repeat** step 26 and 27, but set the **detectionTime** to 12.5		
37	Send an **UpdateObjects** method for a subscription ID other than noted in step 2. The update changes the **validityDuration** to 5		
38	**Verify** a result is returned in which the error code is set to **ProtocolErrorCode** 9 (InvalidObjectReference)		
39	**Verify** that the **validityDuration** is still set to 200		
40	**Verify** that DENM messages are broadcasted		
Tested by:		**Date:**	

7.2.6 Use Case: Delete an ItsEvent

7.2.6.1 *Test Case: Terminate an ItsEvent*

Test case:	Terminate an ItsEvent
ID:	RISFI.UC6.1
Objective:	Verify that DENM messages, that all were created through the RIS-FI, can be terminated with the terminateEvents method
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS

STEP	DESCRIPTION	PASS/FAIL	REMARKS/ACTIONS
1	Send a **CreateEvents** with three **ItsEvents** with the following specification: Event 1: • The **eventType** is **adverseWeatherCondition-Adhesion (6)** • The **eventSubType** is 0 • The **validityDuration** is 40 • The **relevanceDistance** is 6000.0 • The **trafficDirection** is **downstreamTraffic** • The **repetitionInterval** is 1000 Event 2: • The **eventType** is **wrongWayDriving (14)** • The **eventSubType** is 2 • The **validityDuration** is 60 • The **relevanceDistance** is 1230.0 • The **trafficDirection** is **downstreamTraffic** • The **repetitionInterval** is 2000 Event 3: • The **eventType** is **hazardousLocation-DangerouseCurve (96)** • The **eventSubType** is 3 • The **validityDuration** is 80 • The **relevanceDistance** is 1500.0 • The **trafficDirection** is **upstreamTraffic** • The **repetitionInterval** is 3000		

	For the events the following also applies: • For **eventPosition** use E1 described in *Table 2: Events reference*, enter the data in WGS84 coordinates • For **traces** use T2 described in *Table 2: Events reference* • For **destinationArea** is: ▪ **centre** is same as **eventPosition** ▪ **majorAxis** is 1000.0 ▪ **minorAxis** is 1000.0 ▪ **angle** is 0.0 ▪ **circular** is true • The **detectionTime** for all events are set to the current time		
2	**Verify** a result is returned with three object IDs for the defined events ID 1: _____ ID 2: _____ ID 3: _____		
3	**Verify** that for each event at least one DENM is send with the corresponding data fields		
4	Send a **TerminateEvent** for the event with ID 1		
5	**Verify** an empty result is received		
6	**Verify** one DENM for the event with ID1 is send with a termination flag		
7	**Verify** event 1 is no longer received after at least one **repetitionInterval**		
8	**Repeat** step 3 till 7 for object ID 2		
9	**Repeat** step 3 till 7 for object ID 3		
Tested by:		**Date:**	

7.2.6.2 *Test Case Exception: Update or terminate with wrong object ID*

Test case:	Update or terminate with wrong object ID
ID:	RISFI.UC6.2
Objective:	Verify no changes are made to transmitted DENM messages when an **UpdateObjects** method is received for a non-existing objectID. Also verify no DENM messages are terminated when a **terminateEvent** method is received for a non-existing objectID..
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS

STEP	DESCRIPTION	PASS/FAIL	REMARKS/ACTIONS
1	Create a **CreateEvents** with an **ItsEvent** with the following specification: Event 1: • The **eventType** is **hazardousLocation-ObstacleOnTheRoad (10)** • The **eventSubType** is 0 • The **validityDuration** is 60 • The **relevanceDistance** is 400.0 • The **trafficDirection** is **upstreamTraffic** • **For eventPosition** use **E1** described in *Table 2: Events reference* • For **traces** use trace **T1** described in *Table 2: Events reference* • The **detectionTime** is current time • The **repetitionInterval** is 1000 Send the **CreateEvents** method to the RIS		
2	**Verify** a result is returned with a subscription ID for the defined event ID 1: _____		
3	Wait at least 5 seconds before sending an **UpdateObjects** changing the **eventType** to **collisionRisk (97)** for a non-existing object ID		
4	**Verify** a result is returned in which the error code is set to **ProtocolErrorCode** 9 (InvalidObjectReference)		
5	**Verify** the **eventType** is not updated in the transmitted DENM		
6	Send a **TerminateEvent** using a non-existing object ID		
7	**Verify** a result is returned in which the error code is set to **ProtocolErrorCode** 9 (InvalidObjectReference)		
8	**Verify** that no events are terminated and that all DENM messages are transmitted		

Tested by:		Date:	

7.2.7 Use Case Monitoring of events

7.2.7.1 *Test Case: Monitor ItsEvents*

Test case:	ItsEvent updates upon subscription
ID:	RISFI.UC7.1
Objective:	Verify a notification is sent to all ITS applications that have a subscription to **ItsEvents,** after an event has been created, updated or terminated
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS

STEP	DESCRIPTION	PASS/FAIL	REMARKS/ACTIONS
1	Subscribe to ITS events by sending a **SubscribeObjects** method and subscribing to **RISObjectType** of type **ItsEvent**		
2	**Verify** a result is returned with a subscription id **Note** the received **subscription** ID 1: _____		
3	Generate a DENM message on the network with the following minimum specifications: • The **eventType** is **vehicleBreakdown (91)** • The **eventSubType** is 0 • The **validityDuration** is 15 • The **relevanceDistance** is 400.0 • The **trafficDirection** is **upstreamTraffic** • **For eventPosition is** use **E1** described in *Table 2: Events reference* • For **traces** use trace **T1** described in *Table 2: Events reference* • The **detectionTime** is current time		
4	**Verify** that a **NotifyObjects** is received with data corresponding to the generated DENM message		
5	**Verify** the correct object ID is received		
6	**Verify** that the DENM message is logged		
7	Resend the DENM of step 3 with the **relevanceDistance** set to 300		
8	**Verify** that a **NotifyObjects** is received with data corresponding to the generated DENM message		
9	**Verify** the correct object ID is received		
10	**Verify** that the DENM message is logged		

11	Resend the DENM of step 3 with the terminate flag set		
12	**Verify** that a **NotifyObjects** is received with data corresponding to the generated DENM message		
13	**Verify** the correct object ID is received		
14	**Verify** that the DENM message is logged		
15	**Verify** that the event object is removed from the LDM		
Tested by:		**Date:**	

7.2.7.2 *Test Case: Monitoring of ItsEvents using subscription filtering*

Test case:	Monitoring of ItsEvents using subscription filtering		
ID:	RISFI.UC7.2		
Objective:	Verify a notification is sent to the ITS application for an **ItsEvent** that is based on the configured filtering, also after an update of a DENM message.		
Pre-conditions	The RIS is up and running The ITS-CRA is registered to the RIS		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Generate three DENM for events on the network with the following specifications DENM1: • The **StationId** is 2000 • The **eventType** is **accident (2)** • The **eventSubType** is 0 • The **validityDuration** is 30 • The **relevanceDistance** is 500.0 • The **trafficDirection** is **upstreamTraffic (1)** • **For eventPosition is** use **E1** described in *Table 2: Events reference* • For **traces** use trace **T1** described in *Table 2: Events reference* • The **detectionTime** is current time DENM2: • The **StationId** is 2001 • The **eventType** is **roadworks (3)** • The **eventSubType** is **majorRoadworks (1)** • The **validityDuration** is 50 • The **relevanceDistance** is 500.0 • The **trafficDirection** is **upstreamTraffic (1)** • **For eventPosition is** use **E2** described in *Table 2: Events reference* • For **traces** use trace **T3** described in *Table 2: Events reference* • The **detectionTime** is current time		

	DENM3: • The **StationId** is 2002 • The **eventType** is **accident (2)** • The **eventSubType** is **0** • The **validityDuration** is 20 • The **relevanceDistance** is 500.0 • The **trafficDirection** is **oppositeTraffic (3)** • **For eventPosition is** use **E3** described in *Table 2: Events reference* • For **traces** use trace **T4** described in *Table 2: Events reference* • The **detectionTime** is current time		
2	Subscribe to ITS events by sending a **SubscribeObjects** method and subscribing to **ItsEvent** of type **RISObjectType** using the filter "**eventType** = 2". Request that only **eventType** and **eventPosition** are reported.		
3	**Verify** a result is returned with a subscription id and that the notification **NotifyObjects** of the generated DENM messages is received only for DENM1 and DENM3.		
4	**Verify** that only the **eventType** and **eventPosition** of the corresponding stations are reported and that all data corresponds to the generated DENM.		
5	Change the **validityDuration** of DENM2 and resend the DENM message		
6	**Verify** that no **NotifyObjects** is received		
7	Change the **validityDuration** of DENM3 and resend the DENM message		
8	**Verify** a **NotifyObjects** is received for the generated DENM3 message		
9	**Verify** that only the **eventType** and **eventPosition** of the corresponding station is reported and are corresponding to the generated DENM.		
10	Deregister from the RIS		
Tested by:		**Date:**	

7.2.8 Use Case Inform on the signalling status

7.2.8.1 *Test Case: Inform on the signalling status*

Test case:	Inform on the signaling status		
ID:	RISFI.UC8.1		
Objective:	Verify a **SignalGroup** object can be updated as a whole with the **UpdateObjects** method. The **SignalGroup** object can be retrieved through the **NotifyObject** method and the updated **SignalGroup** data is present in the related SPAT message. Verify MAP content.		
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	**Verify** SPAT messages are transmitted between every 1 and 10 seconds with the **IntersectionStatusObject,** set to the value **noValidSPATisAvailableAtThisTime (13)**		
2	**Verify** MAP messages are broadcasted every hour containing the intersection configuration of the RIS **Verify** the MAP message objects match the intersection configuration (see 10 Appendix C)		
3	Subscribe to **SignalGroup** events by sending a **SubscribeObjects** method with object type **SignalGroup (4)**		
4	**Verify** a result is returned containing only a subscription id and the current status of the signalgroups		
5	Send an **UpdateObjects** method with an update for **IntersectionState** with the following specifications: • The **Ids** is intersection ID (TINT1): ▪ The **manualControlIsEnabled** is false, ▪ The **stopTimeIsActivated** is false, ▪ The **failureFlash** is false, ▪ The **preemptIsActive** is false, ▪ The **signalPriorityIsActive** is false, ▪ The **fixedTimeOperation** is false, ▪ The **trafficDependentOperation** is **true**, ▪ The **standbyOperation** is false, ▪ The **failureMode** is false, ▪ The **off** is false		

6	Send an **UpdateObjects** method with an update for **SignalGroup** with a ticks value clearly different from the ticks value received in the Alive message from the RIS itself with the following minimum specifications: • With the following specifications ▪ **Id** is (TINT1_fc02) ▪ The **state** is **PermissiveMovementAllowed (5)** ▪ The **predictions are** as follows ▪ [0] ▪ **state** is **PermissiveMovementAllowed (5)** ▪ **minEnd** is UpdateObjects.Ticks + 20000 ▪ **maxEnd** is UpdateObjects.Ticks + 20300 ▪ **likelyEnd** is UpdateObjects.Ticks + 20200 ▪ **confidence** is 50 ▪ **next** is UpdateObjects.Ticks + 40100 ▪ [1] ▪ **state** is **PermissiveClearance (7)** ▪ **startTime** is UpdateObjects.Ticks + 20200 ▪ **minEnd** is UpdateObjects.Ticks + 40000 ▪ **maxEnd** is UpdateObjects.Ticks + 40300 ▪ **likelyEnd** is UpdateObjects.Ticks + 40100 ▪ **confidence** is 50 ▪ The **validityDuration** is 15 ▪ The **speedProfiles** are as follows ▪ [0] ▪ The **type** is greenwave(1) ▪ The **distance** is 30 ▪ The **speed** is 13.8 ▪ [1] ▪ The **type** is greenwave(1) ▪ The **distance** is 100 ▪ The speed is 19.4 ▪ The **reason** is weather ▪ **Id** is (TINT1_fc03) ▪ The **state** is **StopAndremain (3)** ▪ The **predictions** are as follows		

	▪ **state** is **StopAndremain** (3) ▪ **minEnd** is UpdateObjects.Ticks + 15000 ▪ The **validityDuration** is 15 ▪ **Id** is (TINT1_fc05) ▪ The **state** is **StopAndremain (3)** ▪ The **predictions** are as follows ▪ **state** is **StopAndremain (3)** ▪ **minEnd** is UpdateObjects.Ticks + 25100 ▪ **maxEnd** is null ▪ **likelyEnd** is UpdateObjects.Ticks + 25300 ▪ **confidence** is 50 ▪ The **validityDuration** is 15 ▪ **Id** is (TINT1_fc07) ▪ The **state** is PermissiveMovementAllowed **(5)** ▪ The **predictions** are as follows ▪ **state** is **PermissiveMovementAllowed (5)** ▪ **minEnd** is UpdateObjects.Ticks + 20000 ▪ **maxEnd** is UpdateObjects.Ticks + 20300 ▪ **likelyEnd** is UpdateObjects.Ticks + 20100 ▪ **confidence** is 50 ▪ The **validityDuration** is 15 ▪ The **speedProfiles** are as follows ▪ The **type** is ecoDrive(2) ▪ The **distance** is 20 ▪ The **speed** is 30.5 ▪ **Id** is (TINT1_fc08) ▪ The **state** is PermissiveMovementAllowed **(5)** ▪ The **predictions** are as follows ▪ **state** is **PermissiveMovementAllowed (5)** ▪ **minEnd** is UpdateObjects.Ticks + 20000 ▪ **maxEnd** is UpdateObjects.Ticks + 20030 ▪ **likelyEnd** is null ▪ **Confidence** is null ▪ The **validityDuration** is 15 ▪ The **speedProfiles** as follow,		

	▪ The **type** is none(0) ▪ The **distance** is 0 ▪ The **speed** is 0.0 ▪ **Id** is (TINT1_21) ▪ The **state** is **StopAndremain (5)** ▪ The **predictions** are as follows ▪ **state** is **StopAndremain (5)** ▪ **minEnd** is UpdateObjects.Ticks + 30000 ▪ The **validityDuration** is 15 ▪ **Id** is (TINT1_31) ▪ The **state** is **StopAndremain (5)** ▪ The **predictions** are as follows ▪ **state** is **Unavailable (0)** ▪ **minEnd** is null ▪ The **validityDuration** is 15 Ticks value in the **UpdateObjects** method: _____ Timestamp in the **UpdateObjects** method: _____		
7	**Verify** an empty result is returned		
8	**Verify** the response is received within 100 ms		
9	**Verify** that the received SignalGroup **Notification** corresponds to the previous SignalGroup update		
10	**Verify** SPAT messages are transmitted with the data corresponding to the data in the **SignalGroup** update (see 11 APPENDIX D: SPAT Message (JSON format))		
11	**Verify** that, after the **validityDuration** has expired**,** SPAT messages are transmitted with signalgroup states are set to unavailable, no predictions, no speedprofiles.		
Tested by:		**Date:**	

7.2.8.2 *Test Case: Remove optional fields in SignalGroup*

Test case:	Remove optional fields in **SignalGroup**
ID:	RISFI.UC8.2
Objective:	Verify that the optional fields of the object **SignalGroup** can be removed with an **UpdateObjects** method First the optional field **speedProfil**and is set to a value, then it is removed with the **UpdateObjects** method.
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS

STEP	DESCRIPTION	PASS/FAIL	REMARKS/ACTIONS
1	Subscribe to **SignalGroup** events by sending a **SubscribeObjects** method with object type **SignalGroup (4)**		
2	**Verify** a result is returned containing only a subscription id		

3	Send an **UpdateObjects** method with an update for **SignalGroup** with the following specifications:
	
	• **Id** is (TINT1_fc05)
	• The **state** is StopAndremain **(3)**
	• The **predictions** are as follows
	  ▪ **state** is **StopAndremain (3)**
	  ▪ **minEnd** is UpdateObjects.Ticks + 18200
	  ▪ **maxEnd** is UpdateObjects.Ticks + 19300
	  ▪ **likelyEnd** is UpdateObjects.Ticks + 18310
	  ▪ **confidence** is 40
	  ▪ **next** is UpdateObjects.Ticks + 18450
	• The **validityDuration** is 60
	• The **speedProfiles** are as follows
	  ▪ The **type** is none
	  ▪ The **distance** is 30
	  ▪ The **speed** is 13.8
	• The **reason** is emergencyVehiclePriority(2)		

4	**Verify** an empty result is returned		
5	**Verify** SPAT messages are transmitted with the data corresponding to the data in the **SignalGroup** object (see 11 Appendix D)		

6	**Verify** a **signalGroup** notification is received		
7	Send an **UpdateObjects** method with an update for **SignalGroup** with the following specifications: • **Id** is (TINT1_fc05) ▪ The **speedProfiles** is set to null or an empty list ▪ The **reason** set to null This will remove the optional fields		
8	**Verify** an empty result is returned		
9	**Verify** SPAT messages are transmitted and the **speedProfile** is removed (see 11 Appendix D)		
10	**Verify** a **signalGroup** notification is received without **speedProfile** and without **reason**		
Tested by:		**Date:**	

7.2.8.3 *Test Case: Update Intersection status*

Test case:	Update Intersection status
ID:	RISFI.UC8.3
Objective:	Verify the status field of an Intersection object can be updated with the method **UpdateObjects** and that the intersection object can be retrieved with the **RequestObject** method and the updated Intersection status is present in the related MAP message.
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS

STEP	DESCRIPTION	PASS/FAIL	REMARKS/ACTIONS
1	Send an **UpdateObjects** method with an update for **IntersectionState** with the following specifications: • The **Id** is intersection ID (TINT1) • The **manualControlIsEnabled** is false, • The **stopTimeIsActivated** is false, • The **failureFlash** is false, • The **preemptIsActive** is false, • The **signalPriorityIsActive** is false, • The **fixedTimeOperation** is false, • The **trafficDependentOperation** is true, • The **standbyOperation** is false, • The **failureMode** is false, • The **off** is false		
2	**Verify** an empty **updateObjects** result is returned		
3	**Verify** SPAT messages are transmitted with the updated Intersection data (see 11 Appendix D).		
4	**Repeat** step 1 till 4 but set **manualControlIsEnabled** to true		
5	**Repeat** step 1 till 4 but set **stopTimeIsActivated** to true		
6	**Repeat** step 1 till 4 but set **failureFlash** to true		
7	**Repeat** step 1 till 4 but set **preemptIsActive** to true		
8	**Repeat** step 1 till 4 but set **signalPriorityIsActive** to true		
9	**Repeat** step 1 till 4 but set **fixedTimeOperation** to true		
10	**Repeat** step 1 till 4 but set **trafficDependentOperation** to true		

11	**Repeat** step 1 till 4 but set **standbyOperation** to true		
12	**Repeat** step 1 till 4 but set **failureMode** to true		
13	**Repeat** step 1 till 4 but set **off** to true		
Tested by:		**Date:**	

7.2.8.4 *Test Case Exceptions: Check validity of fields SignalGroup*

Test case:	Check validity of fields **SignalGroup**		
ID:	RISFI.UC8.4		
Objective:	Verify that the RIS checks for mandatory fields missing, field types and field values		
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Send an **UpdateObjects** method with an update for **SignalGroup** with the following specifications: • **Id** is (TINT1_fc02) • The **state** is **PermissiveMovementAllowed (5)** • The **validityDuration** is 600 • The **predictions** are as follows ▪ **state** is **PermissiveMovementAllowed (5)** ▪ **minEnd** is UpdateObjects.Ticks + 20000		
2	**Verify** an empty result is returned		
3	Resend the **UpdateObjects** from step 1 but omit but omit the **predictions** field **minEnd**		
4	**Verify** that the RIS facilities sends in response a message with the error code 6 *(MissingAttribute) (minEnd is missing)*		
5	**Verify** that the original value (in this case 20000) is still present in the SPAT		
6	Send an **UpdateObjects** method with an update for **SignalGroup** with the following specifications: • **Id** is (TINT1_fc05) • The **state** is ProtectedMovementAllowed (6)		

	• The **predictions** are as follow
	  ▪ **state** is ProtectedMovementAllowed (6)
	  ▪ **startTime** is 10233
	  ▪ **minEnd** is UpdateObjects.Ticks + 18200
	  ▪ **maxEnd** is UpdateObjects.Ticks + 19300
	  ▪ **likelyEnd** is UpdateObjects.Ticks + 10000
	  ▪ **confidence** is 40
	  ▪ **next** is UpdateObjects.Ticks + 18450
	• The **validityDuration** is 1200
	• The **speedProfiles** are as follows,
	  ▪ The **type** is none
	  ▪ The **distance** is 30
	  ▪ The **speed** is 13.8
	• The **reason** is publicTransportPriority(1)
	
	Record the resulting SPAT!		

7	**Verify** an empty result is returned		
8	Resend the **UpdateObjects** from step 6 but set the **validityDuration** to 86401		
9	**Verify** a result is returned in which the error code is set to **ProtocolErrorCode** 8 (**InvalidAttributeValue**)(**validityDuration** is out of range) and that the application disconnects		
10	**Repeat** step 8 through 9, but set **state** to 12		
11	**Repeat** step 8 through 9, but set the **predictions** field **state** to 12		
12	**Repeat** step 8 through 9, but set the **predictions** field **startTime** to 4294967296		
13	**Repeat** step 8 through 9, but set the **predictions** field **maxEnd** to 4294967296		
14	**Repeat** step 8 through 9, but set the **predictions** field **minEnd** to 4294967296		
15	**Repeat** step 8 through 9, but set the **predictions** field **likelyEnd** to 4294967296		
16	**Repeat** step 8 through 9, but set the **predictions** field **next** to 4294967296		
17	**Repeat** step 8 through 9, but set the **predictions** field **confidence** to 101		
18	**Repeat** step 8 through 9, but set the **speedProfiles** field **distance** to 429496729.6		
19	**Repeat** step 8 through 9, but set the **speedProfiles** field **speed** to 99.1		
20	**Repeat** step 8 through 9, but set the **speedProfiles** field **type** to 4		

21	**Repeat** step 8 through 9, but set the **reason** to 13		
22	Resend the **UpdateObjects** from step 6 but set **validityDuration** to 200.2		
23	**Verify** a result is returned in which the error code is set to **ProtocolErrorCode** 7 (**InvalidAttributeType**) and that the application disconnects.		
24	**Repeat** step 22 through 23, but set **state** to 12		
25	**Repeat** step 22 through 23, but set the **predictions** field **state** to "5"		
26	**Repeat** step 22 through 23, but set the **predictions** field **startTime** to 100.2		
27	**Repeat** step 22 through 23, but set the **predictions** field **maxEnd** to true		
28	**Repeat** step 22 through 23, but set the **predictions** field **minEnd** to 1.001		
29	**Repeat** step 22 through 23, but set the **predictions** field **likelyEnd** to false		
30	**Repeat** step 22 through 23, but set the **predictions** field **next** to "1200"		
31	**Repeat** step 22 through 23, but set the **predictions** field **confidence** to 4.0		
32	**Repeat** step 22 through 23, but set the **speedProfiles** field **distance** to "30"		
33	**Repeat** step 22 through 23, but set the **speedProfiles** field **speed** to "88"		
34	**Repeat** step 22 through 23, but set the **speedProfiles** field **type** to false		
35	**Repeat** step 22 through 23, but set the **reason** to 1.0		
36	**Verify** the first SPAT received after step 35 is not changed as a result of the programmings starting from step 8 compared to the SPAT recorded in step 6		
Tested by:		**Date:**	

7.2.8.5 *Test Case Exceptions: Unauthorized update of SignalGroup*

Test case:	Unauthorized update of **SignalGroup**		
ID:	RISFI.UC8.5		
Objective:	Verify that an unauthorized user cannot update the **SignalGroup**		
Pre-conditions	The RIS is up and running The ITS-CRA is registered to the RIS		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
7	Send an **UpdateObjects** method with an update for **SignalGroup** with the following specifications: • **Id** is TINT1_fc02 • The **state** is **PermissiveMovementAllowed (5)**		

	• The predictions are as follows ▪ **state** is **PermissiveMovementAllowed (5)** ▪ **minEnd** is UpdateObjects.Ticks + 20000 ▪ **maxEnd** is UpdateObjects.Ticks + 20030 ▪ **likelyEnd** is UpdateObjects.Ticks + 15000 ▪ **confidence** is 50 ▪ **next** is UpdateObjects.Ticks + 20100 • The **speedProfiles** as follow, ▪ The **type** is none ▪ The **distance** is 30 ▪ The **speed** is 13.8		
8	**Verify** a result is returned in which the error code is set to **ProtocolErrorCode** of 2006 or 1 (Not authorized for this object) and that the application is disconnected.		
9	**Verify** the SPAT is not changed		
10	Deregister from the RIS		
Tested by:		**Date:**	

7.2.9 Additional Case: Method RequestObjects

7.2.9.1 *Test Case: Verify CAM objects LDM attributes*

Test case:	Verify CAM objects LDM attributes		
ID:	GENFI.UC9.1		
Objective:	Verify that when a CAM is sent to the RIS, the method **requestObjects** can return the **ItsStation** object based on the CAM and the **ItsStation** object of the RIS itself. The CAM is respectively sent by the following vehicle roles: default, publicTransport, specialTransport, dangerousgoods, roadwork, rescue, emergency, safetyCar.		
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS and has no subscriptions		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Generate a vehicle CAM message with the following specifications:		

	Use the default CAM values from section 5.3.4. • The **StationID** is 3001 • Positioned on lane 1 towards the SG fc02 after the start of the lane path		
2	Send a **RequestObjects** method for **type ItsStation (1)**		
3	**Verify** a result is returned containing the CAM data generated in step 1		
4	Generate a vehicle CAM message like the one in step 1, but with the following altered or added specifications: • The **stationType** is **specialVehicles (10)** • The **StationID** is 3002 • The **vehicleRole** is **publicTransport (1)** • The **EmbarkationStatus** is true • For **PtActivation** ▪ **ptActivationType** is (3) ▪ **ptActivationData** contains (RIS Attributes are shown) ▪ **lineNr** is 34000 ▪ **vehicleID** is 33999 ▪ **serviceNr** is 33998 ▪ **journeyNr** is 33997 ▪ **supportNr** is 33996 ▪ **companyNr** is 129 ▪ **Occupancy** is 33995		
5	Send a **RequestObjects** method with **filter** for type **ItsStation**		
6	**Verify** a result is returned containing multiple CAM messages including the CAM generated at step 4		
7	**Verify** all corresponding generated data fields are present in the **ItsStation** of **StationID** 3002		
8	Generate a CAM message on the network like the one in step 1, but with the following altered or added specifications:		

	• The **stationType** is **specialVehicles (10)**
	• The **StationID** is 3003
	• The **vehicleRole** is **specialTransport (2)**
	• For **RoleAttributes**
	  ▪ **heavyLoad** is true
	  ▪ **excessWidth** is false
	  ▪ **excessLength** is true
	  ▪ **excessHight** is false
	  ▪ **lightBarActivated** is true
	  ▪ **sirenActivated** is true		

9	Send a **RequestObjects** method with **filter** for type **ItsStation**		
10	**Verify** a result is returned containing multiple CAM messages including the CAM generated at step 9		
11	**Verify** all generated data fields are present in the **ItsStation** of **StationID** 3003		
12	Generate a CAM message on the network like the one in step 1, but with the following altered or added specifications: • The **stationType** is **specialVehicles (10)** • The **StationID** is 3004 • The **vehicleRole** is **dangerousGoods (3)** • The **dangerousGoodsBasic** is **oxidizingSubstances** (13)		
13	Send a **RequestObjects** method with **filter** for type **ItsStation**		
14	**Verify** a result is returned containing multiple CAM messages including the CAM generated at step 13		
15	**Verify** all generated data fields are present in the **ItsStation** of **StationID** 3004		
16	Generate a CAM message on the network like the one in step 1 but with the following altered or added specifications: • The **stationType** is **specialVehicles (10)** • The **StationID** is 3005 • The **vehicleRole** is **roadwork (4)**		

	• For **RoleAttributes** ▪ **lightBarActivated** is true		
17	Send a **RequestObjects** method with **filter** for type **ItsStation**		
18	**Verify** a result is returned containing multiple CAM messages including the CAM generated at step 16		
19	**Verify** all generated data fields are present in the **ItsStation** of **StationID** 3005		
20	Generate a CAM message on the network like the one in step 1 but with the following altered or added specifications: • The **stationType** is **specialVehicles (10)** • The **StationID** is 3006 • The **vehicleRole** is **rescue (5)** • For **RoleAttributes** ▪ **sirenActivated** is true		
21	Send a **RequestObjects** method with **filter** for type **ItsStation**		
22	**Verify** a result is returned containing multiple CAM messages including the CAM generated at step 21		
23	**Verify** all generated data fields are present in the **ItsStation** of **StationID** 3006		

24	Generate a CAM message on the network like the one in step 1 but with the following altered or added specifications:
	• The **stationType** is **specialVehicles (10)**
	• The **StationID** is 3007
	• The **vehicleRole** is **emergency (6)**
	• For **RoleAttributes**
	  ▪ **sirenActivated** is true
	  ▪ **lightBarActivated** is true		

25	Send an **RequestObjects** method with **filter** for **type ItsStation**		

26	**Verify** a result is returned containing multiple CAM messages including the CAM generated at step 25		
27	**Verify** all generated data fields are present in the **ItsStation** of **StationID** 3007		
28	Generate a CAM message on the network like the one in step 1 but with the following altered or added specifications: • The **stationType** is **specialVehicles (10)** • The **StationID** is 3008 • The **vehicleRole** is **safetyCar (7)** • For **RoleAttributes** ▪ **sirenActivated** is true ▪ **lightBarActivated** is true		
29	Send an **RequestObjects** method with **filter** for **type ItsStation**		
30	**Verify** a result is returned containing multiple CAM messages including the CAM generated at step 29		
31	**Verify** all generated data fields are present in the **ItsStation** of **StationID** 3008		
Tested by:		**Date:**	

7.2.9.2 *Test Case: Verify unavailable CAM fields are set to null*

Test case:	Verify unavailable CAM fields are set to null
ID:	GENFI.UC9.2
Objective:	Verify that when a CAM is sent to the RIS with unavailable indicators for acceleration, heading, vehicleLength, vehicleWidth and speedValue, that the method requestObjects can return an ItsStation object with null in the fields length, width, heading, speed and acceleration
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS

STEP	DESCRIPTION	PASS/FAIL	REMARKS/ACTIONS
1	Generate a vehicle CAM message with the following specifications: • Positioned on lane 1 towards the signal group (SG) fc02 just after the start of the lane path • The **stationType** is **passengerCar (5)** • The **vehicleRole** is **default (0)** • The **StationID** is 4001 • for **heading** ▪ **headingValue** is **unavailable** (3601) ▪ **headingConfidence** is **unavailable** (127) • for **speed** ▪ **speedValue** is **unavailable** (16383) ▪ **speedConfidence** is **unavailable** (127) • **driveDirection** is **unavailable** (2) • for **vehicleLength** ▪ **vehicleLengthValue** is **unavailable** (1023) ▪ **vehicleLengthConfidenceIndication** is **unavailable(4)** • for **vehicleWidth** is **unavailable** (62) • for **longitudinalAcceleration** ▪ **longitudinalAccelerationValue** is **unavailable** (161) ▪ **longitudinalAccelerationConfidence** is **unavailable** (102) • for **curvature** ▪ **curvatureValue** is **unavailable**(30001) ▪ **curvatureConfidence** is **unavailable** (7)		

	• for **curvatureCalculationMode** is **yawRateUsed** (0)
	• for **yawRate**
	  ▪ **yawRateValue** is **unavailable(32767)**
	  ▪ **yawRateConfidence** is **unavailable** (8)		

2	Send a **RequestObjects** method with **filter** for type **ItsStation**		
3	**Verify** an **ItsStation** result is returned containing the CAM data received **Verify** the **ItsStation** object has **null** for the fields length, width, heading, speed and acceleration		
Tested by:		**Date:**	

7.2.9.3 *Test Case: Filtering ItsStations*

Test case:	Filtering ItsStations
ID:	GENFI.UC9.3
Objective:	Verify correct **ItsStations** are returned with a **requestObjects** when filtering is specified
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS

STEP	DESCRIPTION	PASS/FAIL	REMARKS/ACTIONS
1	Generate three vehicle CAM messages with the following specifications: Vehicle 1: • Positioned on lane 1 towards the signal group (SG) fc02, just after the start of the lane path • The **stationType** is **passengerCar (5)** • The **vehicleLengthValue** is **45 (4,5 m)** • The **role** is **default (0)** Vehicle 2: • Positioned on lane 8 towards the SG 8 just after the start of the lane path		

	• The **stationType** is **heavyTruck (8)** • The **role** is **default (0)** • The **vehicleLengthValue** is **120 (12 m)** • The **heading** is pointing towards the signal group Vehicle 3: • Positioned on lane 9 • The **stationType** is **motorcycle (5)** • The **vehicleLengthValue** is **25 (2,5 m)** • The **role** is **safetyCar (7)** • The **heading** is away from the intersection		
2	Send a **RequestObjects** method filtering for lengths greater than 4 m (length > 4.0)		
3	**Verify** a result is returned containing the CAM data received from the **heavyTruck** and the **passengerCar**		

4	Send a **RequestObjects** method filtering for lengths greater than 4 m (length > 4.0) and smaller than 5 m (length < 5.0)
	
	"type": 1,
	"selection": {
	"attribute": "length",
	"value": 4.0,
	"comparator": ">"
	},
	"and":{
	"attribute": "length",
	"value": 5.0,
	"comparator": "<"
	},
	"report": ["stationType", "speed"]		

5	**Verify** a result is returned containing the data of the **passengerCar.**		
6	**Verify** only the **stationType** and the **speed** of the **passengerCar** are returned		
Tested by:		**Date:**	

7.2.9.4 *Test Case: Request RISFacilities object*

Test case:	Request RISFacilities object		
ID:	GENFI.UC9.4		
Objective:	Verify a **RISFacilities** object can be retrieved with the method **RequestObjects**.		
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Send a **RequestObjects** method with **filter** for type **RISFacilities**		
2	**Verify** a result is returned containing the fields **id**, **location**, **info** and **intersections**		
3	**Verify** the fields match the configuration of the RIS		
Tested by:		**Date:**	

7.2.9.5 *Test Case: Request an ItsEvent*

Test case:	Request an ItsEvent		
ID:	GENFI.UC9.5		
Objective:	Verify an **ItsEvent** can be retrieved with the method **RequestObjects**		
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Send a **CreateEvents** with the following specification: • The **eventType** is **accident** • The **eventSubType** is 0 • The **validityDuration** is 15 • The **relevanceDistance** is 400.0 • The **trafficDirection** is **upstreamTraffic**		

	• **For eventPosition is** use **E1** described in *Table 2: Events reference* • For **traces** use trace **T1** described in *Table 2: Events reference* • The **detectionTime** is current time		
2	Send a **RequestObjects** method with **filter** for type **ItsEvent**		
3	**Verify** a result is returned containing the data created at step 1		
Tested by:		**Date:**	

7.2.9.6 *Test Case: Request partial ItsEvent*

Test case:	Request partial ItsEvent
ID:	GENFI.UC9.6
Objective:	Verify a selection can be made using filters in a **RequestObjects** method and verify that the requested report objects are presented.
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS

STEP	DESCRIPTION	PASS/FAIL	REMARKS/ACTIONS
1	Send a **CreateEvents** with two **ItsEvents** with the following specification: Event 1: • The **eventType** is **hazardousLocation-SurfaceCondition (9)** • The **eventSubType** is 0 • The **validityDuration** is 15 • The **relevanceDistance** is 400.0 • The **trafficDirection** is **upstreamTraffic** • The **repetitionInterval** is 2222 • For **eventPosition** use E1 described in *Table 2: Events reference* • For **traces** use T2 described in *Table 2: Events reference* • Event 2: • The **eventType** is **roadworks hazardousLocation-AnimalOnTheRoad (11)** • The **eventSubType** is 1 • The **validityDuration** is 10 • The **relevanceDistance** is 600.0 • The **trafficDirection** is **upstreamTraffic** • The **repetitionInterval** is 1034 • For **eventPosition** use E3 described in *Table 2: Events reference* • For **traces** use T4 described in *Table 2: Events reference*		
2	Send a **RequestObjects** method with **filter** for type **ItsEvent** and **eventType** is **hazardousLocation-SurfaceCondition (9)** and report the **relevanceDistance** and **eventPosition**		

3	**Verify** a result is received containing only event 1		
4	**Verify** the object returned containing only the **relevanceDistance** and **eventPosition**		
5	Deregister from the RIS		
Tested by:		**Date:**	

7.2.9.7 *Test Case Exception: Change RISFacilities object*

Test case:	Change RISFacilities object
ID:	GENFI.UC9.7
Objective:	Verify a **RISFacilities** object cannot be changed
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS

STEP	DESCRIPTION	PASS/FAIL	REMARKS/ACTIONS
1	Send an **UpdateObjects** method with an update for **RISFacilities** with the following specifications • The **Ids** is "DYN_3399" • The **location** is “Basicweg” • The **intersections** "16" • For **info** ▪ The **fiVersion** is {2,1,1} ▪ The **companyname** is “BLAHBLAH” ▪ The **facilitiesVersion** is {"202"}		
2	**Verify** a result is returned in which the error code is set to **ProtocolErrorCode** 2006 or 1 (Not authorized for this object)		
3	Send a **RequestObjects** method with **filter** for type **RISFacilities**		
4	Verify the **requestObjects** result does not show any of the changes from step 1.		
5	Deregister from the RIS		
6	Register ITS-PRA to the RIS		
7	Repeat step 1		
8	**Verify** a result is returned in which the error code is set to **ProtocolErrorCode** 2006 or 1 (Not authorized for this object)		
9	Send a **RequestObjects** method with **filter** for type **RISFacilities**		

10	**Verify** a result is returned containing the fields **id**, **location**, **info** and **intersections**		
11	**Verify** the fields match to the configuration of the RIS		
Tested by:		**Date:**	

7.2.9.8 *Test Case Exception: Update an ItsStation object*

Test case:	Update an ItsStation object
ID:	GENFI.UC9.8
Objective:	Verify an **ItsStation** cannot be updated, because it is read only
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS

STEP	DESCRIPTION	PASS/FAIL	REMARKS/ACTIONS
1.	Generate a vehicle CAM message with the following specifications: Vehicle 1: • Positioned on lane 1 towards the signal group (SG) 2 just after the start of the lane path • The **stationType** is **passengerCar (5)** • The **role** is **default (0)**		
2.	Send a **RequestObjects** method with **filter** for **type ItsStation**		
3.	**Verify** an **ItsStation** result is returned containing the CAM data, note the object ids: _____		
4.	Send an **UpdateObjects** method with an update for **ItsStation** with the following specifications • The **Ids** is as noted in step 3 • The **role** is 3		

5.	**Verify** a result is returned in which the error code is set to **ProtocolErrorCode** 2006 or 1 (Not authorized for this object)		
6.	Send a **RequestObjects** method with **filter** for **type ItsStation**		
7.	**Verify** an **ItsStation** result is returned containing the CAM data as present in step 1		
Tested by:		**Date:**	

7.2.9.9 *Test Case Exception: Wrong parameters provided*

Test case:	Wrong parameters provided		
ID:	GENFI.UC9.9		
Objective:	Verify that the RIS Facilities takes appropriate actions when a wrong parameter is entered in a method		
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Generate a vehicle CAM message with the following specifications: • Positioned on lane 1 towards the signal group (SG) 2 just after the start of the lane path • The **stationType** is **passengerCar (5)** • The **role** is **default (0)**		

2	Send a **RequestObjects** method with the following parameters
	`"params": {`
	`"type": 2,`
	`"ids": [ "71004_5" ]`
	`}`		

3	**Verify** a result is returned in which the error code is set to **ProtocolErrorCode** 32602 (Invalid params)		
Tested by:		**Date:**	

7.2.9.10 *Test Case Exception: Communication after deregistration*

Test case:	Communication after deregistration		
ID:	GENFI.UC9.10		
Objective:	Verify that after a deregister it is not possible to still communicate with the RIS		
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Deregister the ITS-PRA from the RIS		
2	Send a **RequestObjects** object with **filter** for type **RISFacilities** with the unregistered ITS-PRA		
3	**Verify** a result is returned in which the error code is set to **ProtocolErrorCode** 1 (NotAuthorized) or no result is returned at all.		
Tested by:		**Date:**	

7.2.9.11 ***Test Case: Request Intersection object***

Test case:	Request Intersection object		
ID:	GENFI.UC9.11		
Objective:	Verify an **Intersection** object can be retrieved with the method **RequestObjects**.		
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Send a **RequestObjects** method with **filter** for type **Intersection**		
2	**Verify** a result is returned containing: **id** is ITF controlData section, element “name” in “controlledIntersection” **name** is ITF controlData section, element “descriptive name” in “controlledIntersection” **referencePosition** is ITF refPoint **speedLimit** is ITF regulatorySpeedLimit **lanes[]** is all ITF LaneList->GenericLane **enabledLanes[]** is empty		

	signalGroups[] is list of signalGroup ids **status** is list of Boolean variables which are all false		
3	**Verify** the fields match the configuration of the RIS		
Tested by:		**Date:**	

7.3 Generic-FI

7.3.1 Use Case: Establish connection with the Facilities

7.3.1.1 *Test Case: Basic Session without TLS*

If the user accounts are not yet configured, execute *7.2.1.1 Test Case: Start up and user configuration RIS*

This test case is optional as secure communication is mandatory and unsecure not

Test case:	Basic Session handling without TLS		
ID:	GENFI.UC1.1		
Objective:	Verify a non TLS can be created and maintained		
Pre-conditions	The RIS is up and running		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Connect to the not secured RSU port (see *Table 3: Default network protocol ports*) and register as an ITS-CRA		
2	**Verify** a **RegistrationReply** is received containing the fields **sessionid**, **facilities** and **version**		
3	**Verify** that that all values are correct		
4	**Verify** the keep alive mechanism functions properly		
5	**Verify** the **AliveObject** contains a tick value and a timestamp		
6	**Verify** the tick value is in milliseconds		
7	**Verify** the timestamp is in UTC time and in milliseconds and based on ISO8601		
8	Deregister the application		
9	**Verify** a **DeregistrationReply** is received		
Tested by:		**Date:**	

7.3.1.2 *Test Case: Basic Session handling secure*

Test case:		Basic Session handling secure		
ID:		GENFI.UC1.2		
Objective:		Verify a TLS session can be created and maintained		
Pre-conditions		The RIS is up and running		
STEP	**DESCRIPTION**		**PASS/FAIL**	**REMARKS/ACTIONS**
1	Connect to the secured RSU port (TLS) (see *Table 3: Default network protocol ports*) and register as an ITS-CRA			
2	**Verify** a **RegistrationReply** is received with fields **sessionid**, **facilities** and **version**			
3	**Verify** a that that all values are correct			
4	**Verify** the keep alive mechanism functions properly			
5	**Verify** the **AliveObject** contains a tick value and a timestamp			
6	**Verify** the tick value is in milliseconds			
7	**Verify** the timestamp is UTC in in milliseconds and based on ISO8601			
8	Deregister the application			
9	**Verify** a **DeregistrationReply** is received			
Tested by:			**Date:**	

7.3.2 Use Case: Break connection with the Facilities

7.3.2.1 *Test Case: Subscription ending*

Test case:	Subscription ending
ID:	GENFI.UC2.1
Objective:	Verify all subscriptions of an application are revoked once it deregisters or when the RIS restarts
Pre-conditions	The RIS is up and running

STEP	DESCRIPTION	PASS/FAIL	REMARKS/ACTIONS
1	Generate a DENM message on the with following specifications: • The **eventType** is **vehicleBreakdown (91)** • The **eventSubType** is 0 • The **validityDuration** is 10 • The **relevanceDistance** is 400.0 • The **trafficDirection** is **upstreamTraffic (1)** • **For eventPosition is** use **E1** described in *Table 2: Events reference* • For **traces** use trace **T1** described in *Table 2: Events reference* • The **detectionTime** is current time		
2	Register to the RIS using the **Consumer2** credentials		
3	**Verify** a **RegistrationReply** is received containing the fields **sessionid**, **facilities** and **version**		
4	**Verify** that the **sessionid** is unique		
5	**Verify** that the audit log contains a message that the user has been registered		
6	Subscribe to ITS events by sending a **SubscribeObjects** method and subscribing to **ItsEvent** of type **RISObjectType** and **report = {"eventType", "eventPosition"}**		
7	**Verify** a result is returned with a subscription id		
8	Resend the DENM of step 1		
9	**Verify** a notification is received and only **eventType** and **eventPosition** are reported		
10	Deregister the application		
11	**Verify** a **DeregistrationReply** is received		
12	Resend the DENM of step 1		
13	**Verify** no notification is received		
14	Register again to the RIS using the **Consumer2** credentials		
15	**Verify** a **RegistrationReply** is received		

16	**Verify** that the **sessionid** is unique		
17	Resend the DENM of step 1		
18	**Verify** no notification is received		
19	Subscribe to ITS events by sending a **SubscribeObjects** method and subscribing to **ItsEvent** of type **RISObjectType** and **report = {"eventType", "eventPosition"}**		
20	**Verify** a result is returned with a subscription id		
21	Resend the DENM of step 1		
22	**Verify** a notification is received and only **eventType** and **eventPosition** are reported		
23	Disconnect from the RIS by removing the network connection. Reconnect after 30 seconds		
24	Resend the DENM of step 1		
25	**Verify** no notification is received as the connection is broken due to a keep alive timeout		
26	**Verify** that the audit log contains a message that the user has been deregistered		
27	Register again to the RIS using the **Consumer2** credentials		
28	**Verify** a **RegistrationReply** is received		
29	**Verify** that the **sessionid** is unique		
30	**Verify** that the audit log contains a message that the user has been registered		
31	Resend the DENM of step 1		
32	**Verify** no notification is received as no subscription has been placed after registration		
33	Subscribe to ITS events by sending a **SubscribeObjects** object and subscribing to **ItsEvent** of type **RISObjectType** and **report = {"eventType", "eventPosition"}**		
34	**Verify** a result is returned with a subscription id		
35	Resend the DENM of step 1		
36	**Verify** a notification is received and only **eventType** and **eventPosition** are reported		
37	Restart the RIS and wait for it to be restarted		
38	Resend the DENM of step 1		
39	**Verify** no notification is received		
40	Register again to the RIS using the **Consumer2** credentials		
41	**Verify** a **RegistrationReply** is received		
42	**Verify** that **sessionid** is unique		
43	Resend the DENM of step 1		
44	**Verify** no notification is received		
45	Subscribe to ITS events by sending a **SubscribeObjects** object and subscribing to **ItsEvent** of type **RISObjectType** and **report = {"eventType", "eventPosition"}**		

46	**Verify** a result is returned with a subscription id		
47	Resend the DENM of step 1		
48	**Verify** a notification is received and only **eventType** and **eventPosition** are reported		
49	Deregister the application		
50	**Verify** a **DeregistrationReply** is received		
Tested by:		**Date:**	

7.3.3 Use Case: Revoke ITS Application authorization

7.3.3.1 *Test Case: Revoke ITS Application authorization*

Test case:	Revoke ITS Application authorization		
ID:	GENFI.UC4.1		
Objective:	Verify that the RIS Facilities can revoke the authorization of a ITS-A		
Pre-conditions	The RIS is up and running An ITS application is registered		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Force the RIS to revoke the authorisation of the ITS-A		
2	**Verify** that the ITS-A receives a deregistration message		
3	**Verify** that the ITS-A gets disconnected and the session terminated		
4	**Verify** that the ITS-A cannot update any data from any object and neither receive updates		
Tested by:		**Date:**	

7.3.4 Use Case: Exception Handling

7.3.4.1 *Test Case: Illegal login credentials*

Test case:	Illegal login credentials		
ID:	GENFI.UC4.1		
Objective:	Verify the RIS rejects illegal logons when using the credentials of *Table 8: Credential test case login*		
Pre-conditions	The RIS is up and running		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Configure the Session details of the RIS-FI in the ITS-A with the credentials of *Table 8: Credential test case login* below **User**: <User> **Password**: <Password> **Application type**: <Type>		
2	Activate the connect and register procedure		
3	**Verify** that a result is returned in which the error code is set to **ProtocolErrorCode** 1 (NotAuthorized)		
4	**Verify** the socket is closed.		
5	**Verify** that the audit log contains the failed connect and register event		
6	Repeat step 1 until 5 for all usernames of the table		
Tested by:		**Date:**	

User	Password	Type	Comment
Control1	Im?h@ppy!2meet(<you>)&5isSpeci@l	Control	Application type not used in RIS-FI
	WeNeedEnoughUsers!10min	Provider	No username
Provider	Pr.v.d.r1	Provider	Wrong username
Provider1	Wrong_password	Provider	Wrong password
Provider	My.p@ssw0rd4	Provider	Wrong user-password combination

Consumer1		Consumer	No password
Consumer1	Password1	3	Wrong specified application type

Table 8: Credential test case login

7.3.4.2 *Test Case: Illegal second logon*

Test case:	Illegal second logon		
ID:	GENFI.UC4.2		
Objective:	Verify that the RIS Facilities only allows one session per valid username		
Pre-conditions	The RIS is up and running		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1.	Activate the connect and register procedure for a ITS-PRA using the credentials for **Provider1**		
2.	**Verify** that the ITS-PRA is connected and registered to the RIS-FI		
3.	**Verify** that the audit log contains a message that the user has been registered		
4.	Activate the connect and register procedure for a second ITS-PRA using the exact same credentials for **Provider1**		
5.	**Verify** that **ProtocolErrorCode** 1 (NotAuthorized) is received		
6.	**Verify** the error situation is logged		
7.	**Verify** the session is terminated and that the socket is closed.		
8.	**Verify** that ITS-PRA1 remains connected, registered and active		
Tested by:		**Date:**	

7.3.4.3 *Test Case: RIS-FI fails to send keep alive messages*

Test case:	RIS-FI fails to send keep alive messages		
ID:	GENFI.UC4.3		
Objective:	Verify that the RIS Facilities takes correct actions when no keep alive message is received		
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	The ITS-PRA sends keep alive messages		

2	**Verify** that the connection between the RIS and the ITS-PRA remains active		
3	Force the ITS-PRA to stop sending keep alive messages for 18 seconds Send a **RequestObjects** method with **filter** for type **RISFacilities**		
4	**Verify** that the connection between the RIS and the ITS application remains active		
5	**Verify** a result is returned		
6	Continue to force the ITS-PRA to stop sending keep alive messages, but to stay connected		
7	**Verify** that the RIS facilities keeps sending keep alive messages		
8	**Verify** that after 25 seconds (2.5 times the interval of 10s for an ITS-CRA or PRA) the ITS-PRA gets disconnected and the session terminated		
9	**Verify** that the audit log contains a message that the user has been disconnected		
10	Send a **RequestObjects** method with **filter** for type **RISFacilities**		
11	**Verify** it is not possible to send the object to the RIS		
Tested by:		**Date:**	

7.3.4.4 *Test Case: Handling of message bursts*

Test case:	Handling of message bursts		
ID:	GENFI.UC4.4		
Objective:	Verify the RIS can handle message bursts and assure the order is handled properly		
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	The ITS-PRA sends a message burst consisting of 10 messages with a time span of 100 ms between each message.		
2	**Verify** that the RIS Facilities handles the messages in the proper order (through the time ticks or the time stamp in the log file of the exerciser)		
3	The ITS-PRA sends a message burst consisting of 100 messages with a time span of 100 ms between each message		
4	**Verify** that the RIS Facilities handles the messages in the proper order (through the time ticks or the time stamp in the log file of the exerciser)		
Tested by:		**Date:**	

7.3.4.5 *Obsolete*

7.3.4.6 *Test Case: ITS-A sends no registration request*

Test case:	ITS-A sends no registration request		
ID:	GENFI.UC4.6		
Objective:	Verify that an ITS-A gets disconnected if it connects, but doesn't send a registration request		
Pre-conditions	The RIS is up and running		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	An ITS-PRA connects to the RIS-FI, but doesn't send a registration request		
2	**Verify** that the RIS Facilities terminates the session with the ITS-PRA after the alive timeout period (2.5 times the interval of 10 seconds is 25 seconds) is expired		
Tested by:		**Date:**	

7.3.4.7 ***Test Case:*** *Registration within active session*

Test case:	*Registration within active session*		
ID:	GENFI.UC4.7		
Objective:	Verify that the RIS Facilities can handle a registration within an active session		
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	The same ITS-PRA sends a registration request to the RIS-FI		
2	**Verify** that **ProtocolErrorCode** 1 (NotAuthorized) is received		
3	**Verify** the error situation is logged		
4	**Verify** the session is terminated and that the socket is closed.		
Tested by:		**Date:**	

7.3.4.8 *Test Case: Use of incompatible protocol*

Test case:		Use of incompatible protocol		
ID:		GENFI.UC4.8		
Objective:		Verify that the RIS Facilities takes appropriate actions when an application using an incompatible protocol tries to connect.		
Pre-conditions		The RIS is up and running		
STEP	**DESCRIPTION**		**PASS/FAIL**	**REMARKS/ACTIONS**
1	Try to connect with an IVERA client.			
2	**Verify** the RIS disconnects the connection			
3	Register to the RIS as ITS-PRA			
4	**Verify** a registration reply is received.			
Tested by:			**Date:**	

7.3.4.9 *Test Case: Handling of newer protocol version*

Test case:		Handling of newer protocol		
ID:		GENFI.UC4.9		
Objective:		Verify that the RIS Facilities can handle an application using a newer protocol version		
Pre-conditions		The RIS is up and running		
STEP	**DESCRIPTION**		**PASS/FAIL**	**REMARKS/ACTIONS**
1	An ITS-PRA connects and registers to the RIS using a newer protocol version then the RIS itself is using			
2	**Verify** that the RIS Facilities accepts the connection of the ITS-PRA.			
Tested by:			**Date:**	

7.3.4.10 *Test Case: Handling of older (un-supported) protocol version*

Test case:		Handling of older protocol		
ID:		GENFI.UC4.10		
Objective:		Verify that the RIS Facilities can handle an application using an older (un-supported) protocol version		
Pre-conditions		The RIS is up and running ITS-PRA is available with an older (un-supported) protocol version		
STEP	**DESCRIPTION**		**PASS/FAIL**	**REMARKS/ACTIONS**

1	The ITS-PRA tries to connect and register itself to the RIS		
2	**Verify** that the RIS Facilities detects this situation and reports this explicitly back to the ITS-A		
Tested by:		**Date:**	

7.3.4.11 *Test Case: Unknown method*

Test case:	Unknown method		
ID:	GENFI.UC4.11		
Objective:	Verify that the RIS sends a proper respond when an unsupported (or undefined) method is received		
Pre-conditions	The RIS is up and running The ITS-CLA is registered to the RIS		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Send the following unsupported method to the RIS: `{"jsonrpc": "2.0", "method": "foo" "params": "bar"}`		
2	**Verify** that the RIS-FI sends in response a message with the error code 32601 (*method not found) (*when a response is requested)		
Tested by:		**Date:**	

7.3.4.12 *Test Case: Invalid and large JSON message*

Test case:	Invalid and large JSON message		
ID:	GENFI.UC4.12		
Objective:	Verify that the RIS Facilities correctly handles invalid of large JSON messages		
Pre-conditions	The RIS is up and running The ITS-PRA is registered to the RIS		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Send an **UpdateObjects** method with an update for **SignalGroup** with the following specifications: • **Id** is (TINT1_fc02) • The **state** is **PermissiveMovementAllowed (5)** • The **validityDuration** is 600 • The **predictions** are as follows ▪ **state** is **PermissiveMovementAllowed (5)** ▪ **minEnd** is 20000 Append to the following JSON message to the update **SignalGroup** message: • {"jsonrpc": "2.0", "method": "foobar, "params": "bar", "baz] Send the message to the RIS		
2	**Verify** the session is disconnected and no result is returned		
3	**Verify** SPAT messages are broadcasted without the signalgroup data of step 1.		
4	**Verify** that the audit log contains the deregistration of the ITS-PRA		
5	Reconnect and register the ITS-PRA again to the RIS		
6	Create a single JSON message exceeding the specified size of the receive buffer of the RIS by concatenating multiple alive messages Send the message to the RIS		
7	**Verify** the session is disconnected and no result is returned		
8	**Verify** that the audit log contains the deregistration of the ITS-PRA		
Tested by:		**Date:**	

7.3.4.13 ***Test Case:*** *Deregistration without active session*

Test case:	*Deregistration without active session*		
ID:	GENFI.UC4.13		
Objective:	Verify that the RIS Facilities can handle a deregistration outside an active session		
Pre-conditions	The RIS is up and running No ITS-A is registered to the RIS		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Connect to the RIS and send a deregistration request to the RIS-FI		
2	**Verify** that **ProtocolErrorCode** 0 (Error) is received		
Tested by:		**Date:**	

7.3.5 Additional Case: Performance tests

For the performance tests multiple applications have to connect to the RIS and must be able to communicate concurrent at the same time. Preferable an automated test is used here.

7.3.5.1 *Test Case: Multiple sessions handling*

Test case:	Multiple sessions handling		
ID:	GENFI.UC5.1		
Objective:	Verify the RIS can process 20 concurrent requests / replies each second from 10 ITS Applications Each ITS Application has 10 subscriptions		
Pre-conditions	The RIS is up and running 10 ITS-A are available		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	For each ITS-A create 10 subscriptions to the RIS		
2	**Verify** a **RegistrationReply** is received for each subscription containing the fields **sessionid**, **facilities** and **version**		
3	**Verify** that each **sessionid** is unique		
4	**Verify** the keep alive mechanism functions properly for each subscription		
5	In each subscription, at the same time, send a **RequestObjects** object with **filter** for **type RISFacilities** followed by the same **RequestObjects** object after 500 ms		
6	**Verify** the responses are received within 100 ms		
Tested by:		**Date:**	

7.3.5.2 *Test Case: Multiple session notifications*

Test case:	Multiple session notifications		
ID:	GENFI.UC5.2		
Objective:	Verify 10 ITS applications can receive 25 notifications per second per application		
Pre-conditions	The RIS is up and running 10 ITS applications are registered to the RIS		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Prepare for 25 different vehicles a CAM message with the following specifications: • Positioned on lane 1 towards the signal group 2 just after the start of the lane path • The **stationType** is **passengerCar (5)** • The **role** is **default (0)** • The **speed** value is 1098 (10,98 m/s) Assure 25 different station ID's are used		
2	For each vehicle in step 1, send one CAM message per second for 25 seconds. When doing this, stick to the following rules: ▪ Between each CAM message there is an average interval of 0,04 seconds ▪ Each time a vehicle sends an update, the position of the vehicle is moved 2 meters towards the SG ▪ Each time a vehicle sends an update, increase the **speed** value by 1		
3	**Verify** that each application has received 25 **NotifyObjects** for each ItsStation, meaning it received a total of 625 **NotifyObjects**		
4	**Verify** for 10 random **NotifyObjects** that they contain the correct data corresponding to the generated CAM messages		
5	**Verify** the notifications are received within 500 + <max_latency_CAM> ms after the CAM was sent		
Tested by:		**Date:**	

7.3.5.3 *Test Case: C-ITS message performance*

Test case:	C-ITS message performance		
ID:	GENFI.UC5.3		
Objective:	Verify 250 IC-ITS messages can be processed per second and can lead to an update or additional object. 220 vehicles will drive over the intersection and vehicles will send a SRM message		
Pre-conditions	The RIS is up and running ITS-CRA1 is registered to the RIS as **Consumer1** and ITS-CRA2 is registered to the RIS as **Consumer2**		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
	For 220 vehicles multiple CAM messages (4 positions) have to be prepared with the following specification: • All vehicles have the following specifications (and defaults from § 5.3.4): ▪ The **stationType** is **passengerCar (5)** ▪ The **role** is **default (0)** ▪ The **speed** value is random between 1000 (10.00 m/s) and 2000 (20.0 m/s) • On each of the following lanes 44 vehicles are approaching ▪ Lanes 1, 2, 5, 7, 8 • All the vehicles will follow the following path ▪ **Position 1:** Before the lane (not mapped) ▪ **Position 2:** On the lane (mapped) ▪ **Position 3:** Middle of the intersection between lanes (mapped) ▪ **Position 4:** Over the intersection (mapped) ▪ Lane 1 drives to 10 ▪ Lane 2 drives to 6 ▪ Lane 8 drives to 3 ▪ Lane 7 drives to 6 ▪ For path 5 (SG5) the following accounts ▪ 11 vehicles drive to path 10 ▪ 11 vehicles drive to path 9 ▪ 11 vehicles drive to path 3 ▪ 11 vehicles drive to path 4		

	• For 5 vehicles on each of the 5 lanes generate 4 SRM messages (so 100 in total): ▪ Where **IntersectionReferenceID** is TINT1 ▪ Where **RequestID** is unique for each vehicle ▪ Where **PriorityRequestType** is for each position ▪ Position 1 **priorityRequest (1)** ▪ Position 2 **priorityRequestUpdate (2)** ▪ Position 3 **priorityRequestUpdate (2)** ▪ Position 4 **priorityCancellation (3)** ▪ Where **inBoundLane** is the lane the vehicle is on		
1	Subscribe ITS-CRA1 to PrioritizationRequest by sending a **SubscribeObjects** method and subscribing to **PrioritizationRequest**		
2	**Verify** a result is returned with a subscription id		
3	Subscribe ITS-CRA2 to ITS station events by sending a **SubscribeObjects** method and subscribing to **ItsStation** of type **RISObjectType** with filter condition "**matches** != null"		
4	**Verify** a result is returned with a subscription id		
5	Send the following messages 1 second after each other: • the CAM messages of **Position 1** and the SRM • the CAM messages of **Position 2** and the SRM • the CAM messages of **Position 3** and the SRM • the CAM messages of **Position 4** and the SRM		
6	**Verify** ITS-CRA1 received 100 **NotifyObjects** corresponding to the generated SRM		
7	**Verify** the notifications are received within 500 + <latency_SRM_transmission> ms after generating the SRM		
8	**Verify** ITS-CRA2 received 660 **NotifyObjects** corresponding to the generated CAM messages. (220 do not result in a Notify as these are not matched on the map)		
9	**Verify** each vehicle generated at **Position 2, 3** and **4** is mapped and contain the fields **intersection**, **lane**, **signalgroup**, **distance** and **offset**		
10	**Verify** the notifications are received within 500 + <latency_CAM_transmission> ms after generating the CAM		
Tested by:		**Date:**	

7.3.6 Additional Case: Security

7.3.6.1 *Test Case: Verify TLS Implementation*

Test case:	Verify TLS Implementation		
ID:	GENFI.UC6.1		
Objective:	Verify no weaknesses exist in the TLS implementation		
Pre-conditions	The RIS is up and running The nmap tool is available on the test machine/ testers laptop, the latest version of the OWASP o-saft tool is available on the test machine. (see § 5.2.5 Network tools)		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Use nmap to scan for open tcp and udp ports and note the open ports: …… …… `nmap –sU –sS –sV –reason –PN –n <target IP>` with `<target-ip>` replaced by the IP of the device under test. Note: nmap can take up to 20 minutes to return with a result.		
2	For every port reported in step 1 (except port 22) execute the following o-saft command: `perl o-saft.pl +check –no-dns –v <target-ip>:<port>` with `<target-ip>` replaced by the IP of the device under test and `<port>` replaced with one of the ports found in step 1.		

3	**Verify** for every port:
	- The following ciphers are present:
	  ○ TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
	  ○ TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
	  ○ TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
	  ○ TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
	- There are no weak, low or medium ciphers present		

Tested by:		**Date:**	

7.3.6.2 *Test Case: Storing of security settings*

Test case:	Verify the security settings are stored in a secure way		
ID:	GENFI.UC6.2		
Objective:	Verify the security settings (passwords, certificates) cannot be obtained from the RIS using an insecure measure		
Pre-conditions	The RIS is up and running		
STEP	**DESCRIPTION**	**PASS/FAIL**	**REMARKS/ACTIONS**
1	Ask the vendor the following information: • where and how in the RIS the security settings are stored • how can they be edited • which interface has to be used to do so		
2	**Verify** that the security settings are stored in an encrypted format		
3	**Verify** the security settings are not accessible via an interface that does not meet the mandatory basic security requirements (encrypted)		
4	**Verify** that the security settings shall be accessible to administrators only		
Tested by:		**Date:**	

8 APPENDIX A: Requirements traceability.

This section provides a statement of the compliance of this test specification with the following specifications:

- 🕮 2 IRS RIS-FI
- 🕮 2 Generic-FI
- 🕮 4 Koppelvlakken iVRI Overnamepunt
- 🕮 5, iTLC Architecture (not covered in IRS or Generic)
- 🕮 9 IRS Security v1.1

A list of sections in this document in which the requirement is supported is listed and a comment describing the compliance statement.

Requirement	Compliance	Sections	Comments
IRSIDD_RISFI_GEN_001	C	7.3.1.1	
IRSIDD_RISFI_GEN_002	C	7.2.4.2	
IRSIDD_RISFI_PROT_001	C	7.3.1.1	
IRSIDD_RISFI_PROT_002	C	7.2, 7.3	
IRSIDD_RISFI_PROT_003	C	7.2.1.8, 7.2.7, 7.3.2.1, 7.3.5.2	
IRSIDD_RISFI_PROT_004	N		IP communication is asynchronous. No need to do further testing.
IRSIDD_RISFI_SEC_001	C	7.2.4.5, 7.2.5.3, 7.2.8.5, 7.2.9.7, 7.2.9.8	
IRSIDD_RISFI_REG_001	C	7.2.4.5, 7.2.5.3, 7.2.8.5, 7.2.9.7, 7.2.9.8 7.3.1.1, 7.3.1.2	
IRSIDD_RISFI_REG_002	C	7.3.2.1	
IRSIDD_RISFI_REG_003	C	7.3.2.1, 7.3.4.5	

Requirement	Compliance	Sections	Comments
IRSIDD_RISFI_REG_004	C	7.3.4.5, 7.3.4.7	
IRSIDD_RISFI_REG_005	N		Not supported in the IDD, hence not tested.
IRSIDD_RISFI_REG_006	C	7.3.2.1	
IRSIDD_RISFI_REG_007	C	7.2.9.10	
IRSIDD_RISFI_REG_008	N		IDD does not support priority level, hence this is not tested.
IRSIDD_RISFI_LDM_DD_001	C		Verified with all tests that cover the RIS-FI objects
IRSIDD_RISFI_LDM_DD_002	N		IDD does not support priority level, hence this is not tested.
IRSIDD_RISFI_LDM_DD_003	C		Covered by ObjectID fields in the IDD. These are verified in the tests
IRSIDD_RISFI_LDM_DD_004	C	7.3	Mandatory and optional fields are covered in multiple tests
IRSIDD_RISFI_LDM_DD_005	C	7.3	Covered by all tests
IRSIDD_RISFI_LDM_DT_001	P		Covered in multiple tests In vehicle Information not covered.
IRSIDD_RISFI_LDM_DT_002	C	7.2.9.1, 7.2.9.27.2.9.1	
IRSIDD_RISFI_LDM_DT_003	C	7.2.4	
IRSIDD_RISFI_LDM_DT_004	C	7.2.8.1, 7.2.8.3	
IRSIDD_RISFI_LDM_DT_005	C	7.2.3.1, 7.2.8	
IRSIDD_RISFI_LDM_DT_006	N		In vehicle Information not covered.
IRSIDD_RISFI_LDM_DT_007	C	7.2.8	

Requirement	Compliance	Sections	Comments
IRSIDD_RISFI_LDM_DT_008	N		DetectionArea object not present in IDD. No coverage.
IRSIDD_RISFI_LDM_DT_009	N		Items not according to IDD.
IRSIDD_RISFI_LDM_DT_010	N		DrivingLane not according to IDD, therefor not testable.
IRSIDD_RISFI_LDM_DPRV_001	C	7.2.4	
IRSIDD_RISFI_LDM_DPRV_002	C	7.2.5	
IRSIDD_RISFI_LDM_DPRV_003	N		Not supported in the IDD, hence not tested.
IRSIDD_RISFI_LDM_DPRV_004	C	7.2.6.1	
IRSIDD_RISFI_LDM_DCONS_001	C	7.2.2.1,7.2.9.3	
IRSIDD_RISFI_LDM_DCONS_002	C	7.2.2.1,7.2.9.3	
IRSIDD_RISFI_LDM_DCONS_003	N		Ordering data results not supported in IDD, hence no coverage.
IRSIDD_RISFI_LDM_DCONS_004	N		Query of LDM not supported in IDD, hence no coverage
IRSIDD_RISFI_LDM_DCONS_005	N		Query of LDM not supported in IDD, hence no coverage
IRSIDD_RISFI_LDM_DCONS_006	P	7.2.1.4, 7.2.2.1,7.2.9.3 7.2.3.1, 7.2.7.1	No coverage for periodic notifications.
IRSIDD_RISFI_LDM_DCONS_007	C	7.2.1.6	
IRSIDD_RISFI_LDM_DCONS_008	C	7.2.1.4, 7.2.2.1,7.2.9.3 7.2.3.1, 7.2.7.1	

Requirement	Compliance	Sections	Comments
IRSIDD_RISFI_LDM_DCONS_009	N		Query of LDM not supported in IDD, hence no coverage
IRSIDD_RISFI_LDM_DCONS_010	N		IDD does not support priorities for subscriptions: no coverage
IRSIDD_RISFI_LDM_MAINT_001	C	7.2.4.2, 7.2.5.2	
IRSIDD_RISFI_LDM_STOR_001	N		Not covered. Part of ETSI ITS G5 specifications.
IRSIDD_RISFI_LDM_G5_001	C	7.2.4.2, 7.2.4.3, 7.2.6.1, 7.2.6.2	
IRSIDD_RISFI_LDM_G5_002	C	7.2.1.5, 7.2.1.8 ,7.2.2.1, 7.2.3.1	
IRSIDD_RISFI_LDM_G5_003	C	7.2.8.1, 7.2.8.2, 7.3.4.12	
IRSIDD_RISFI_LDM_G5_004	N		In vehicle Information not covered.
IRSIDD_RISFI_LDM_TOPO_001	P	7.2.8.3, 7.2.9.4, 7.2.9.7	Only status of an intersection can be changed according to the IDD.
IRSIDD_RISFI_LDM_TOPO_002	P	7.2.8.3, 7.2.9.4, 7.2.9.7	Only status of an intersection can be changed according to the IDD.
IRSIDD_RISFI_SVC_001	C	7.3.1.1	
IRSIDD_RISFI_SVC_002	P	7.2.8.3, 7.2.9.4, 7.2.9.7	Topology metadata not in the IDD, hence it is not covered in the tests.
IRSIDD_RISFI_QA_SCAL_001	C	7.3.5.1, 7.3.5.2	
IRSIDD_RISFI_QA_SCAL_002	C	7.3.5.1	
IRSIDD_RISFI_QA_SCAL_003	C	7.3.5.1, 7.3.5.2	
IRSIDD_RISFI_QA_SCAL_004	C	7.3.5.2	
IRSIDD_RISFI_QA_PERF_001	C	7.2.4.2,7.2.8.1,7 .3.5.1, 7.3.5.2	

Requirement	Compliance	Sections	Comments
IRSIDD_RISFI_QA_PERF_002	P	7.3.5.3	IDD does not support priorities for subscriptions: no coverage
IRSIDD_RISFI_QA_PERF_003		7.3.5.3	
IRSIDD_RISFI_QA_AVAIL_001	C	7.3.4.3	
IRSIDD_RISFI_QA_AVAIL_002	N		Not supported in IDD
IRSIDD_RISFI_QA_EVO_001	P	7.3.4.10	Older supported version not testable for first iteration
IRS_SEC_GEN_001	N		// No apllicable, private network, external factor
IRS_SEC_GEN_002	N		No applicable
IRS_SEC_GEN_003	N		No applicable
IRS_SEC_GEN_004	Y	7.3	
IRS_SEC_GEN_005	Y	7.3	
IRS_SEC_GEN_006	Y	7.3.2.1	
IRS_SEC_GEN_007	N		No applicable
IRS_SEC_GEN_009	Y	7.2.1.1	
IRS_SEC_GEN_013	N		Not applicable, operational scope
IRS_SEC_GEN_014	N		Not applicable, development scope
IRS_SEC_GEN_015	N		Not applicable, project scope
IRS_SEC_GEN_016	N		Not applicable, operational scope
IRS_SEC_GEN_017	Y	7.3.6.2	
IRS_SEC_GEN_018	Y	7.3.1.1, 7.3.1.2, 7.3.4.1	

Requirement	Compliance	Sections	Comments
IRS_SEC_RIS_001	Y	7.3.1.1, 7.3.1.2, 7.3.4.1	
IRS_SEC_RIS_002	Y	7.3.1.1, 7.3.1.2, 7.3.4.1	
IRS_SEC_RIS_003	Y	7.3.1.2	
IRS_SEC_RIS_004	Y	7.2.1.1,	
IRS_SEC_RIS_005	Y	7.2.1.3, 7.3.1.2	
IRS_SEC_RIS_006	Y	7.3.4.1	
IRS_SEC_RIS_007	N		Wifi-p is not covered in the type test
iTLC Architecture 9.3.4.4, Minimum logging	Y	7.2.1.3, 7.2.1.7, 7.2.2	
iTLC Architecture 9.3.1.1, Authentication, registration-failures	Y	7.3.4.1	
iTLC Architecture 9.3.4.1, Configuration, Management Entity	Y	7.2.1.2	

9 APPENDIX B: Test result overview

Here an overview of all the test cases are given so an overview can be created of the passed and failed tests

Used C-ITS mode:

[ ] Connected [ ] Hybrid Connected

[ ] Cooperative [ ] Hybrid Cooperative

Test Case	Pass / Fail	Notes
7.2.1.1		
7.2.1.2		
7.2.1.3		
7.2.1.4		
7.2.1.5		
7.2.1.6		
7.2.1.7		
7.2.1.8		
7.2.1.9		
7.2.2.1		
7.2.2.2		
7.2.3.1		
7.2.3.2		
7.2.3.3		
7.2.3.4		

7.2.4.1		
7.2.4.2		
7.2.4.3		
7.2.4.4		
7.2.4.5		
7.2.5.1		
7.2.5.2		
7.2.5.3		
7.2.5.4		
7.2.6.1		
7.2.6.2		
7.2.7.1		
7.2.7.2		
7.2.8.1		
7.2.8.2		
7.2.8.3		
7.2.8.4		
7.2.8.5		
7.2.9.1		
7.2.9.2		

7.2.9.3		
7.2.9.4		
7.2.9.5		
7.2.9.6		
7.2.9.7		
7.2.9.8		
7.2.9.9		
7.2.9.10		
7.2.9.11		
7.3.1.1		
7.3.1.2		
7.3.2.1		
7.3.3.1		
7.3.4.1		
7.3.4.2		
7.3.4.3		
7.3.4.4		
7.3.4.6		
7.3.4.7		
7.3.4.8		

7.3.4.9		
7.3.4.10		
7.3.4.11		
7.3.4.12		
7.3.4.13		
7.3.5.1		
7.3.5.2		
7.3.5.3		
7.3.6.1		
7.3.6.2		

10 APPENDIX C: MAPData Message (JSON format)

This MAP representation can be generated using the web-site http://asn1-playground.oss.com/

```
rec1value MAPEM ::=
{
  header
  {
    protocolVersion currentVersion,
    messageID mapem,
    stationID 3399
  },
  map
  {
    msgIssueRevision 0,
    intersections
    {
      {
        name "TINT1",
        id
        {
          region 1,
          id 1
        },
        revision 0,
        refPoint
        {
          lat 521551700,
          long 53872060
        },
        laneWidth 300,
        speedLimits
        {
          {
            type vehicleMaxSpeed,
            speed 347
          }
        },
        laneSet
        {
          {
            laneID 1,
            name "fc02",
            ingressApproach 1,
            laneAttributes
            {
```

```
    directionalUse { ingressPath },
    sharedWith { individualMotorizedVehicleTraffic },
    laneType vehicle : {  }
  },
  nodeList nodes :
    {
      {
        delta node-XY2 :
          {
            x 996,
            y 499
          },
        attributes
        {
          localNode
          {
            stopLine
          }
        }
      },
      {
        delta node-XY5 :
          {
            x 6619,
            y -3
          },
        attributes
        {
          localNode
          {
            divergePoint
          }
        }
      },
      {
        delta node-LatLon :
          {
            lon 53946584,
            lat 521552146
          }
      }
    },
  connectsTo
  {
    {
```

```
      connectingLane
      {
        lane 10,
        maneuver { maneuverStraightAllowed }
      },
      signalGroup 1,
      connectionID 1
    },
    {
      connectingLane
      {
        lane 9,
        maneuver { maneuverStraightAllowed }
      },
      signalGroup 1,
      connectionID 9
    }
  }
},
{
  laneID 2,
  name "fc03",
  ingressApproach 1,
  laneAttributes
  {
    directionalUse { ingressPath },
    sharedWith { individualMotorizedVehicleTraffic },
    laneType vehicle : {  }
  },
  nodeList nodes :
    {
      {
        delta node-XY2 :
          {
            x 996,
            y 118
          },
        attributes
        {
          localNode
          {
            stopLine
          }
        }
      },
```

```
        {
          delta node-XY5 :
            {
              x 5542,
              y 0
            }
        },
        {
          delta node-XY3 :
            {
              x 1077,
              y 380
            }
        }
      },
    connectsTo
    {
      {
        connectingLane
        {
          lane 6,
          maneuver { maneuverLeftAllowed }
        },
        signalGroup 2,
        connectionID 2
      }
    }
  },
  {
    laneID 3,
    name "eg1a",
    egressApproach 1,
    laneAttributes
    {
      directionalUse { egressPath },
      sharedWith { individualMotorizedVehicleTraffic },
      laneType vehicle : {  }
    },
    nodeList nodes :
      {
        {
          delta node-XY2 :
            {
              x 996,
              y -230
```

```
            }
          },
          {
            delta node-XY6 :
              {
                x 9968,
                y 0
              }
          }
        }
    },
    {
      laneID 4,
      name "eg1b",
      egressApproach 1,
      laneAttributes
      {
        directionalUse { egressPath },
        sharedWith { individualMotorizedVehicleTraffic },
        laneType vehicle : {  }
      },
      nodeList nodes :
        {
          {
            delta node-XY2 :
              {
                x 996,
                y -580
              }
          },
          {
            delta node-XY6 :
              {
                x 9968,
                y 0
              }
          }
        }
    },
    {
      laneID 5,
      name "fc05",
      ingressApproach 2,
      laneAttributes
      {
```

```
  directionalUse { ingressPath },
  sharedWith { individualMotorizedVehicleTraffic },
  laneType vehicle : {  }
},
nodeList nodes :
  {
    {
      delta node-XY3 :
        {
          x -489,
          y -1699
        },
      attributes
      {
        localNode
        {
          stopLine
        }
      }
    },
    {
      delta node-XY6 :
        {
          x 0,
          y -29982
        }
    }
  },
connectsTo
{
  {
    connectingLane
    {
      lane 4,
      maneuver { maneuverRightAllowed }
    },
    signalGroup 3,
    connectionID 3
  },
  {
    connectingLane
    {
      lane 9,
      maneuver { maneuverLeftAllowed }
    },
```

```
      signalGroup 3,
      connectionID 4
    },
    {
      connectingLane
      {
        lane 10,
        maneuver { maneuverLeftAllowed }
      },
      signalGroup 3,
      connectionID 10
    },
    {
      connectingLane
      {
        lane 3,
        maneuver { maneuverRightAllowed }
      },
      signalGroup 3,
      connectionID 11
    }
  }
},
{
  laneID 6,
  name "eg2",
  egressApproach 2,
  laneAttributes
  {
    directionalUse { egressPath },
    sharedWith { individualMotorizedVehicleTraffic },
    laneType vehicle : {  }
  },
  nodeList nodes :
    {
      {
        delta node-XY3 :
          {
            x -847,
            y -1699
          }
      },
      {
        delta node-XY6 :
          {
```

```
            x 0,
            y -9994
          }
      }
    }
},
{
  laneID 7,
  name "fc07",
  ingressApproach 3,
  laneAttributes
  {
    directionalUse { ingressPath },
    sharedWith { individualMotorizedVehicleTraffic },
    laneType vehicle : {  }
  },
  nodeList nodes :
    {
      {
        delta node-XY4 :
          {
            x -2273,
            y -580
          },
        attributes
        {
          localNode
          {
            stopLine
          }
        }
      },
      {
        delta node-XY5 :
          {
            x -4984,
            y 0
          }
      },
      {
        delta node-XY2 :
          {
            x -648,
            y 350
          }
```

```
        }
      },
    connectsTo
    {
      {
        connectingLane
        {
          lane 6,
          maneuver { maneuverRightAllowed }
        },
        signalGroup 4,
        connectionID 5
      }
    }
  },
  {
    laneID 8,
    name "fc08",
    ingressApproach 3,
    laneAttributes
    {
      directionalUse { ingressPath },
      sharedWith { individualMotorizedVehicleTraffic },
      laneType vehicle : {  }
    },
    nodeList nodes :
      {
        {
          delta node-XY4 :
            {
              x -2273,
              y -230
            },
          attributes
          {
            localNode
            {
              stopLine
            }
          }
        },
        {
          delta node-XY5 :
            {
              x -5632,
```

```
            y 0
          },
        attributes
        {
          localNode
          {
            divergePoint
          }
        }
      },
      {
        delta node-LatLon :
          {
            lon 53796805,
            lat 521551490
          }
      }
    },
  connectsTo
  {
    {
      connectingLane
      {
        lane 3,
        maneuver { maneuverStraightAllowed }
      },
      signalGroup 5,
      connectionID 6
    },
    {
      connectingLane
      {
        lane 4,
        maneuver { maneuverStraightAllowed }
      },
      signalGroup 5,
      connectionID 12
    }
  }
},
{
  laneID 9,
  name "eg3a",
  egressApproach 3,
  laneAttributes
```

```
    {
      directionalUse { egressPath },
      sharedWith { individualMotorizedVehicleTraffic },
      laneType vehicle : {  }
    },
    nodeList nodes :
      {
        {
          delta node-XY4 :
            {
              x -2273,
              y 118
            }
        },
        {
          delta node-XY6 :
            {
              x -9969,
              y 0
            }
        }
      }
  },
  {
    laneID 10,
    name "eg3b",
    egressApproach 3,
    laneAttributes
    {
      directionalUse { egressPath },
      sharedWith { individualMotorizedVehicleTraffic },
      laneType vehicle : {  }
    },
    nodeList nodes :
      {
        {
          delta node-XY4 :
            {
              x -2273,
              y 499
            }
        },
        {
          delta node-XY6 :
            {
```

```
            x -9969,
            y 0
          }
        }
      }
    },
    {
      laneID 24,
      name "fc21",
      ingressApproach 2,
      laneAttributes
      {
        directionalUse { ingressPath },
        sharedWith { cyclistVehicleTraffic },
        laneType bikeLane : {  }
      },
      nodeList nodes :
        {
          {
            delta node-XY3 :
              {
                x -1635,
                y -969
              },
            attributes
            {
              localNode
              {
                stopLine
              }
            }
          },
          {
            delta node-XY1 :
              {
                x -149,
                y 0
              }
          },
          {
            delta node-XY1 :
              {
                x -60,
                y 0
              }
```

```
},
{
  delta node-XY1 :
    {
      x -109,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -149,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -149,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -149,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -149,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -150,
      y 0
    }
},
{
  delta node-XY1 :
```

```
    {
      x -149,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -149,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -149,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -149,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -149,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -150,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -149,
      y 0
```

```
      }
  },
  {
    delta node-XY1 :
      {
        x -149,
        y 0
      }
  },
  {
    delta node-XY1 :
      {
        x -99,
        y 0
      }
  },
  {
    delta node-XY1 :
      {
        x -99,
        y 0
      }
  },
  {
    delta node-XY1 :
      {
        x -99,
        y 0
      }
  },
  {
    delta node-XY1 :
      {
        x -100,
        y 0
      }
  },
  {
    delta node-XY1 :
      {
        x -99,
        y 0
      }
  },
  {
```

```
  delta node-XY1 :
    {
      x -99,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -99,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -99,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -99,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -99,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -99,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -100,
```

```
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -99,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -99,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -99,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -99,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -99,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -99,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -99,
      y 0
    }
},
```

```
{
  delta node-XY1 :
    {
      x -99,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -100,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -99,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -99,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -99,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -99,
      y 0
    }
},
{
  delta node-XY1 :
    {
```

```
      x -99,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -99,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -99,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -100,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -99,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x -99,
      y 0
    }
},
{
  delta node-XY3 :
    {
      x -1395,
      y 0
    }
```

```
},
{
  delta node-XY1 :
    {
      x 0,
      y -500
    }
},
{
  delta node-XY1 :
    {
      x 0,
      y -499
    }
},
{
  delta node-XY1 :
    {
      x 0,
      y 0
    }
},
{
  delta node-XY1 :
    {
      x 0,
      y -500
    }
},
{
  delta node-XY1 :
    {
      x 0,
      y -469
    }
},
{
  delta node-XY4 :
    {
      x 2606,
      y 0
    }
},
{
  delta node-XY2 :
```

```
    {
      x 996,
      y 0
    }
},
{
  delta node-XY2 :
    {
      x 996,
      y 0
    }
},
{
  delta node-XY2 :
    {
      x 996,
      y 0
    }
},
{
  delta node-XY3 :
    {
      x 1121,
      y 0
    }
},
{
  delta node-XY2 :
    {
      x 0,
      y -999
    }
},
{
  delta node-XY1 :
    {
      x 0,
      y 0
    }
},
{
  delta node-XY2 :
    {
      x 0,
      y -999
```

```
            }
        },
        {
          delta node-XY2 :
            {
              x 0,
              y -998
            }
        },
        {
          delta node-XY2 :
            {
              x 0,
              y -759
            }
        }
      },
    connectsTo
    {
      {
        connectingLane
        {
          lane 29,
          maneuver { maneuverStraightAllowed }
        },
        signalGroup 6,
        connectionID 0
      }
    }
  },
  {
    laneID 29,
    name "egfc21",
    egressApproach 2,
    laneAttributes
    {
      directionalUse { egressPath },
      sharedWith { cyclistVehicleTraffic },
      laneType bikeLane : {  }
    },
    nodeList nodes :
      {
        {
          delta node-XY2 :
            {
```

```
            x 996,
            y -979
          }
      },
      {
        delta node-XY5 :
          {
            x 5094,
            y 0
          }
      },
      {
        delta node-XY3 :
          {
            x 0,
            y -1859
          }
      },
      {
        delta node-XY5 :
          {
            x -5702,
            y 0
          }
      },
      {
        delta node-XY4 :
          {
            x 0,
            y -3857
          }
      }
    }
},
{
  laneID 32,
  name "fc31_1",
  ingressApproach 1,
  egressApproach 1,
  laneAttributes
  {
    directionalUse { ingressPath, egressPath },
    sharedWith { pedestriansTraffic },
    laneType crosswalk : {  }
  },
```

```
    nodeList nodes :
      {
        {
          delta node-XY2 :
            {
              x 597,
              y 809
            },
          attributes
          {
            localNode
            {
              stopLine
            }
          }
        },
        {
          delta node-XY4 :
            {
              x 0,
              y 3497
            }
        }
      },
    connectsTo
    {
      {
        connectingLane
        {
          lane 31,
          maneuver { maneuverStraightAllowed }
        },
        signalGroup 7,
        connectionID 7
      }
    }
  },
  {
    laneID 31,
    name "fc31_2",
    ingressApproach 1,
    egressApproach 1,
    laneAttributes
    {
      directionalUse { ingressPath, egressPath },
```

```
            sharedWith { pedestriansTraffic },
            laneType crosswalk : {  }
          },
          nodeList nodes :
            {
              {
                delta node-XY3 :
                  {
                    x 597,
                    y -1139
                  },
                attributes
                {
                  localNode
                  {
                    stopLine
                  }
                }
              },
              {
                delta node-XY3 :
                  {
                    x 0,
                    y -1159
                  }
              }
            },
          connectsTo
          {
            {
              connectingLane
              {
                lane 32,
                maneuver { maneuverStraightAllowed }
              },
              signalGroup 7,
              connectionID 8
            }
          }
        }
      }
    }
  },
  dataParameters
  {
```

```
      processAgency "Sweco",
      lastCheckedDate "2017-07-17 13:29:08.045000"
    }
  }
}
```

11 APPENDIX D: SPAT Message (JSON format)

```
{
  "header":{
    "protocolVersion":1,
    "messageID":4,
    "stationID":3399          // StationID from the RIS itself
  },
  "spat":{
    "intersections":[
      {
        "name":"TINT1",
        "id":{
          "region":1,
          "id":1
        },
        "revision":1,
        "status":"0000001000000000",
        "moy":480159,         // take arrival time of SPAT in UTC and calc (daynumber – 1) *
                              // 1440 + hh*60 + mm and compare this to moy
        "timeStamp":26088,    // take seconds and milli seconds part of arrival time of spat and
                              // calc ss * 1000 + ms and compare this to timeStamp
        "states":[
          {
            "movementName":"fc02",
            "signalGroup":1,
            "state-time-speed":[
              {
                "eventState":"permissive-Movement-Allowed",
                "timing":{              // see Note 1 and Note 2
                  "minEndTime":23860,
                  "maxEndTime":23863,
                  "likelyTime":23862,
                  "confidence":2,
                  "nextTime":24061
                },
                "speeds":[
                  {
                    "type":"greenwave",
                    "speed":138,
                    "distance":30
                  }
                ],
                "regional":[
                  {
                    "regionId":3
                    "regExtValue":{
                      "stateChangeReason":"weather"
                    }
                  }
                ]
              },
              {
                "eventState":"permissive-clearance",
                "timing":{              // see Note 1 and Note 2
                  "startTime":24061
                  "minEndTime":24260,
                  "maxEndTime":24263,
                  "likelyTime":24261,
```

```
          "confidence":2
        },
        "speeds":[
          {
            "type":"greenwave",
            "speed":194,
            "distance":100
          }
        ]
      }
    ]
  },
  {
    "movementName":"fc03",
    "signalGroup":2,
    "state-time-speed":[
      {
        "eventState":"stop-And-Remain",
        "timing":{               // see Note 1 and Note2
          "minEndTime":23360,
          "likelyTime":36001,
        }
      }
    ]
  },
  {
    "movementName":"fc05",
    "signalGroup":3,
    "state-time-speed":[
      {
        "eventState":"stop-And-Remain",
        "timing":{              // see Note 1 and Noe 2
          "minEndTime":23911,
          "maxEndTime":36001,
          "likelyTime":23913,
          "confidence":2
        }
      }
    ]
  },
  {
    "movementName":"fc07",
    "signalGroup":4,
    "state-time-speed":[
      {
        "eventState":"permissive-Movement-Allowed",
        "timing":{               // see Note 1 and note 2
          "minEndTime":23860,
          "maxEndTime":23863,
          "likelyTime":23861,
          "confidence":2
        },
        "speeds":[
          {
            "type":"greenwave",
            "speed":305,
            "distance":20
          }
        ]
      }
```

```
          ]
        },
        {
          "movementName":"fc08",
          "signalGroup":5,
          "state-time-speed":[
            {
              "eventState":"permissive-Movement-Allowed",
              "timing":{                // see Note 1 and note 2
                "minEndTime":23860,
                "maxEndTime":23860,
                "likelyTime":36001,
                "confidence":15
              },
              "speeds":[
                {
                  "type":"greenwave",
                  "speed":0,
                  "distance":0
                }
              ]
            }
          ]
        },
        {
          "movementName":"21",
          "signalGroup":6,
          "state-time-speed":[
            {
              "eventState":"stop-And-Remain",
              "timing":{                // see Note 1 and note 2
                "minEndTime":23960,
                "likelyTime":36001
              }
            }
          ]
        },
        {
          "movementName":"31",
          "signalGroup":7,
          "state-time-speed":[
            {
              "eventState":"unavailable"
              "timing":{                // see Note 1 and note 2
                "minEndTime":36001,
                "likelyTime":36001,
              }
            }
          ]
        }
      ]
    }
  ]
 }
}
```

Note 1

When startTime is omitted at RIS_Fi side, then startTime in the SPAT can either be omitted or become 36001. When startTime is set to null at RIS_FI side then the startTime in the SPAT must become 36001.

When minEnd is set to null at RIS_FI side then the minEndTime in the SPAT must become 36001.

When maxEnd is omitted at RIS_Fi side, then maxEndTime in the SPAT can either be omitted or become 36001. When maxEnd is set to null at RIS_FI side then the maxEndTime in the SPAT must become 36001.

When likelyEnd is omitted or set to null at RIS_FI side, then likelyTime in the SPAT must become 36001.

When next is omitted at RIS_Fi side, then nextTime in the SPAT can either be omitted or become 36001. When next is set to null at RIS_FI side then the next in the SPAT must become 36001.

[ELSE]

Calculate startTime, minEndTime, maxEndTime, likelyTime and nextTime
in the SPAT the following way:

Use the timestamp (hh:mm:ms) noted in step 6 (ie. The timestamp in the updateObject method) of 7.2.8.1 to calculate the basetime (BTM) in [TimeMark]:

BTM = mm * 600 + ms / 100

Then calc the expected timeMark values in the SPAT based on the corresponding RIS_FI values:

Ticks_A = ticks value in the updateObject method

Ticks_U = ticks value in the SignalGroup update

Spat_Value = (BTM + ((Ticks_U - Ticks_A + 4294967296) % 4294967296) / 100) % 36000

Note 2

When likelyEnd is omitted or set to null at RIS_FI side, confidence in the SPAT is either omitted or set to 15.

When confidence is omitted at RIS_FI side and a valid likelyEnd is present, then confidence in the SPAT must be set to 15.

[ELSE]

Calculate confidence in the SPAT according to 170629 SPAT Profile v1.2 [subWG Dutch Profile] Level 4.5.

12 APPENDIX E: SSM Message (JSON format)

```
rec1value SSEM ::=
{
  header
  {
    protocolVersion currentVersion,
    messageID ssem,
    stationID 3399
  },
  ssm
  {
    timeStamp 491935,
    second 59893,
    sequenceNumber 29,
    status
    {
      {
        sequenceNumber 29,
        id
        {
          region 1,
          id 1
        },
        sigStatus
        {
          {
            requester
            {
              id stationID : 7007,
              request 104,
              sequenceNumber 5,
              typeData
              {
                role publicTransport,
                subrole requestSubRoleUnKnown
              }
            },
            inboundOn connection : 6,
            minute 491935,
            second 59893,
            duration 65000,
            status granted
          }
        }
      }
    }
  }
}
```