ETSI TR 103 099 V1.5.1 (2022-03)

Intelligent Transport Systems (ITS); Architecture of conformance validation framework

Reference

RTR/ITS-213

Keywords

architecture, conformance, ITS, testing

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - APE 7112B
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° w061004871

Important notice

Notice of disclaimer & limitation of liability

The information provided in the present deliverable is directed solely to professionals who have the appropriate degree of experience to understand and interpret its content in accordance with generally accepted engineering or other professional standard and applicable regulations.
No recommendation as to products and services or vendors is made or should be implied.
No representation or warranty is made that this deliverable is technically accurate or sufficient or conforms to any law and/or governmental rule and/or regulation and further, no representation or warranty is made of merchantability or fitness for any particular purpose or against infringement of intellectual property rights.
In no event shall ETSI be held liable for loss of profits or any other incidental or consequential damages.

Any software contained in this deliverable is provided "AS IS" with no warranties, express or implied, including but not limited to, the warranties of merchantability, fitness for a particular purpose and non-infringement of intellectual property rights and ETSI shall not be held liable in any event for any damages whatsoever (including, without limitation, damages for loss of profits, business interruption, loss of information, or any other pecuniary loss) arising out of or related to the use of or inability to use the software.

Copyright Notification

Contents

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The declarations pertaining to these essential IPRs, if any, are publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org/).

Pursuant to the ETSI Directives including the ETSI IPR Policy, no investigation regarding the essentiality of IPRs, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

DECT™, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™** and **LTE™** are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners. **oneM2M™** logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners. **GSM**® and the GSM logo are trademarks registered and owned by the GSM Association.

Foreword

This Technical Report (TR) has been produced by ETSI Technical Committee Intelligent Transport Systems (ITS).

Modal verbs terminology

In the present document "**should**", "**should not**", "**may**", "**need not**", "**will**", "will not", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

1 Scope

The present document provides a description of the architecture of the ITS conformance validation framework, including definition of the test environment, codec and test adapter. It provides, as well, all the necessary source code to build and run the ITS conformance validation framework.

The ITS conformance validation framework integrates the test suites ETSI TS 102 871-3 [i.5], ETSI TS 102 868-3 [i.6], ETSI TS 102 869-3 [i.7], ETSI TS 102 870-3 [i.8], ETSI TS 102 859-3 [i.9] and ETSI TS 103 191-3 [i.10].

2 References

2.1 Normative references

Normative references are not applicable in the present document.

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] Void.

[i.2] ETSI EG 201 015 (V2.1.1): "Methods for Testing and Specification (MTS); Standards engineering process; A handbook of validation methods".

[i.3] IEEE 802.11p™: "IEEE Standard for Local and Metropolitan Area Networks - Specific requirements; Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications; Amendment 6: Wireless Access in Vehicular Environments".

[i.4] ETSI EN 303 613 (V1.1.1): "Intelligent Transport Systems (ITS); LTE-V2X Access layer specification for Intelligent Transport Systems operating in the 5 GHz frequency band".

[i.5] ETSI TS 102 871-3 (V1.5.1): "Intelligent Transport Systems (ITS); Testing; Conformance test specifications for GeoNetworking; Part 3: Abstract Test Suite (ATS) and Protocol Implementation eXtra Information for Testing (PIXIT)".

[i.6] ETSI TS 102 868-3 (V1.5.1): "Intelligent Transport Systems (ITS); Testing; Conformance test specifications for Cooperative Awareness Basic Service (CA); Part 3: Abstract Test Suite (ATS) and Protocol Implementation eXtra Information for Testing (PIXIT)".

[i.7] ETSI TS 102 869-3 (V1.6.1): "Intelligent Transport Systems (ITS); Testing; Conformance test specifications for Decentralized Environmental Notification Basic Service (DEN); Part 3: Abstract Test Suite (ATS) and Protocol Implementation eXtra Information for Testing (PIXIT)".

[i.8] ETSI TS 102 870-3 (V1.2.1): "Intelligent Transport Systems (ITS); Testing; Conformance test specifications for Geonetworking Basic Transport Protocol (BTP); Part 3: Abstract Test Suite (ATS) and Protocol Implementation eXtra Information for Testing (PIXIT)".

[i.9] ETSI TS 102 859-3 (V1.3.1): "Intelligent Transport Systems (ITS); Testing; Conformance test specifications for Transmission of IP packets over Geonetworking; Part 3: Abstract Test Suite (ATS) and Protocol Implementation eXtra Information for Testing (PIXIT)".

[i.10]	ETSI TS 103 191-3 (V1.3.1): "Intelligent Transport Systems (ITS); Testing; Conformance test specifications for Facilities layer protocols and communication requirements for infrastructure services; Part 3: Abstract Test Suite (ATS) and Protocol Implementation eXtra Information for Testing (PIXIT)".
[i.11]	Titan™ project.

NOTE: Available at https://projects.eclipse.org/projects/tools.titan.

[i.12]	Jenkins®.

NOTE: Available at https://www.jenkins.io/.

[i.13]	Doxygen®.

NOTE: Available at https://www.doxygen.nl/index.html.

3 Definition of terms, symbols and abbreviations

3.1 Terms

Void.

3.2 Symbols

Void.

3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

AC	Adapter Control
ACC	Adaptive Cruise Control
API	Application Programming Interface
ASN	Abstract Syntax Notation
AT	Authorization Ticket
ATS	Abstract Test Suite
BTP	Basic Transport Protocol
BTP-A	Basic Transport Protocol - Type A
BTP-B	Basic Transport Protocol - Type B
CA	Cooperative Awareness
CAM	Cooperative Awareness Message
CC	Cruise Control
DEN	Decentralized Environmental Notification
DENM	Decentralized Environmental Notification Message
EN	European Standard
ETH	ETHernet
GN	GeoNetworking
GN6	GeoNetworking over IPv6
GNSS	Global Navigation Satellite System
HB	High Beam
ID	Identity
IP	Internet Protocol
ITS	Intelligent Transportation Systems
ITS-S	Intelligent Transportation Systems - Station
IUT	Implementation Under Test
IVI	Infrastructure to Vehicle Information
IVIM	Infrastructure to Vehicle Information Message

JDK	Java™ Development Kit
LB	Low Beam
LS	Location Service
LT	Left Turn
LTE	Long Term Evolution
MAC	Media Access Control
MAP	MapData
MAPE	Road/lane topology and traffic maneuver
MAPEM	Road/lane topology and traffic maneuver message
MTC	Main Test Component
MTS	Methods for Testing and Specification
OS	Operating System
OSI	Open Systems Interconnection model
PC	Personal Computer
PDF	Portable Document Format
PDU	Protocol Data Unit
PICS	Protocol Implementation Conformance Statement
PKI	Public Key Infrastructure
RLT	Road and Lane Topology
RSU	Road Side Unit
RT	Right Turn
RTCM	Radio Technical Commission for Maritime services
RTCMEM	RTCM Extended Message
SHB	Single Hop Broadcast
SPaT	Signal Phase and Timing
SPATEM	Signal Phase And Timing Message
SREM	Signal Request Message
SSEM	Signal Request Status Message
SUT	System Under Test
TA	Test Adapter
TCI	TTCN-3 Control Interface
TLM	Traffic Light Maneuver
TSB	Topology Scoped Broadcast
TSS&TP	Test Suite Structure and Test Purposes
TTCN-3	Testing and Test Control Notation 3
UDP	User Datagram Protocol
UdpIp	User datagram protocol/Internet protocol
UT	Upper Tester
UTC	Universal Time Coordinated
V2X	Vehicule to Any

4 Test platform overview

4.1 Constraints and requirements

The purpose of the ITS test platform is to provide a reliable set of software and hardware equipment that can be used to validate TTCN-3 Abstract Test Suites (ATS) developed in ETSI.

The architecture of this test platform has been designed taking into account the following constraints:

- to be compatible with the requirements expressed in the validation handbook (ETSI EG 201 015 [i.2]);
- to be independent of the platform used to implement the test system;
- to be independent of the TTCN-3 tool provider;
- to be configurable and customizable;
- to provide tools and well defined interfaces to System Under Test (SUT), allowing test automation;

- to be easily extensible for future ITS protocols;
- to provide generic components that can be reused in other test platforms.

In order to ensure independence of hardware platforms, all software pieces running on the test platform have been implemented using Java™ language, using generic and widely used libraries.

NOTE: Java™ is the trade name of a programming language developed by Oracle Corporation. This information is given for the convenience of users of the present document and does not constitute an endorsement by ETSI of the programming language named. Equivalent programming languages may be used if they can be shown to lead to the same results.

Test tool independence has been achieved by isolating the tool specific interfaces from core functionalities of the platform. Adapting the current platform to a different test tool would only require the implementation of a very simple piece of software mapping tool-specific functions to generic functions defined in this project.

In addition, great care has been taken to separate ITS specific functionalities from generic test platform tasks in order to provide a maximum number of reusable components for future test platforms.

4.2 General architecture

Typically a TTCN-3 test platform is composed of four different components:

- The TTCN-3 test tool providing necessary software to execute the abstract test suites.
- The hardware equipment supporting TTCN-3 test execution and adaptation to SUTs.
- The codecs which convert protocol messages into their abstract TTCN-3 representation.
- The Test Adapter (TA) implementing interfaces with the device under test.

The interaction of these components is described in Figure 1.

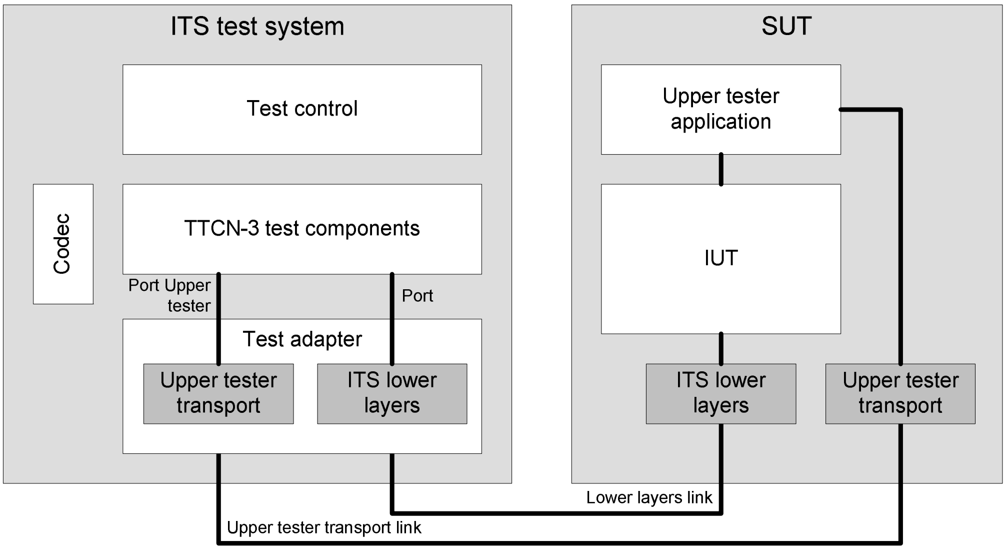


Figure 1: General architecture

The TTCN-3 test tools are usually provided by third parties and their description is out of the scope of the present document. The implementation details of the other components are described in the present document.

5 ITS Test System requirements

5.1 Hardware

The main hardware component of the ITS test platform is a standard PC. Its role is to host the execution of the test suites using a TTCN-3 test execution tool.

Whatever operating system is installed on the computer, it is necessary to ensure that the following points are taken into account:

- No firewall interference with traffic generated by the Test System and/or SUT.
- Excellent time synchronization between the SUT and the test system.
- Test system processes (especially the test adapter) need to be granted unrestricted control to telecommunication hardware.

Time synchronization is maybe the most critical point to be checked before starting any test session, as it can be the source of unpredictable SUT behaviour and generate incoherent results. Indeed, most ITS protocol messages feature a time tag used by the receiver to determine if the information it carries is still valid; if the test system is ahead in time, all messages it sends will be considered either as coming from the future or from a very old date, and be discarded.

This PC is equipped with two network cards, one being used for ITS communication with SUT (lower layers link), the other one being used for exchanging upper tester messages (upper tester transport link). Separating these two communications on different hardware interfaces is not an absolute necessity, but it is a good practice and it ensures that there will be no interaction between the flows.

The communication between the SUT and the test system is achieved through Ethernet if the SUT supports it or using an access layer adaptation box, as shown in Figure 2 and in Figure 3.

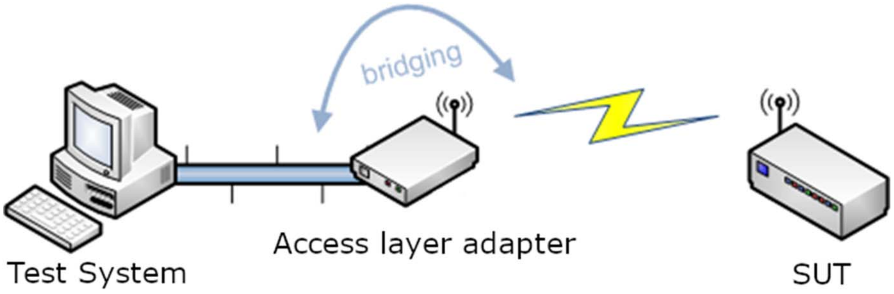


Figure 2: Communication via access layer adaptation box

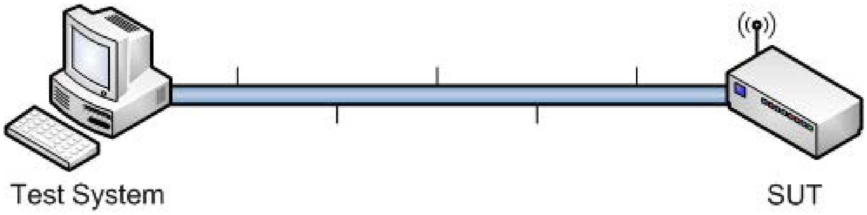


Figure 3: Communication via Ethernet

5.2 Access layer adaptation

The ITS protocol stack makes use of various access layer radio protocols in order to establish communication between ITS devices. For the moment the test suite has been tested using following access layers:

- The ITS G5 radio protocol (IEEE 802.11p [i.3]).
- The LTE/C-V2X protocol (ETSI EN 303 613 [i.4]).

To achieve G5 or C-V2X connectivity, a dedicated hardware equipment needs to be added to the test platform. The role of this access layer adapter is to handle all radio-related tasks transparently and to act as a bridge for the test system, as depicted in Figure 2.

5.3 Software

The ETSI ITS Test System is based on TITAN™ project [i.11] and its core components. The ETSI ITS Test System requires a UNIX®/Linux®-like environment.

NOTE 1: UNIX® is a registered trademark of The Open Group.

NOTE 2: Linux® is the registered trademark of Linus Torvalds in the U.S. and other countries.

5.4 Virtualization

A dockerized version of the ETSI ITS Test System is available, but it is recommended to send and receive ITS messages over UDP.

5.5 Continuous Integration

Located at the root of the source code architecture, a script name *.jenkins.sh* is provided in order to integrate the ETSI Test System source code in a Continuous Integration mechanism based on Jenkins® [i.12].

5.6 Code documentation

Based on Doxygen® [i.13], a documentation in PDF can be generated.

6 Codecs

6.1 Introduction

The codec entity is responsible for the encoding and decoding of TTCN-3 abstract values into bitstrings suitable to be sent to the System Under Test (SUT).

In order to simplify implementation and to ease the maintenance, coding and decoding tasks are handled by several codecs:

- One independent codec package per tested protocol.
- One codec package for TTCN-3 types that do not correspond to real protocol messages. It includes for example all auxiliary types used to carry information to/from Test Adapter, like the ones defined in Test System modules (CamInd, CamReq, etc.).
- One generic codec package available for handling default codec operation non related to any specific protocol. Theses codecs will be used if no protocol-specific codec exists for one type.

6.2 Advanced details of implementation

Each codec is responsible for correctly encoding and decoding one specific type and implements the `codec` interface (e.g. cam_codec implement encoding and decoding for ETSI ITS CAM protocol).

Selection of correct codec for encoding or decoding a message at runtime is managed by:

- the `type` of the message;

- the `encoding` as specified in TTCN-3 modules using "with encode" statements.

7 Test Adapter

7.1 Introduction

The test adapter conceptually splits into three parts:

- a lower test adapter;
- a TTCN-3 platform adapter implementing timers;
- an upper test adapter.

7.2 Lower Tester

7.2.1 Overview

TTCN-3 test suites are usually focusing on a single protocol layer and designed to be executed against real implementations (IUT). However, it is unusual to find standalone implementations as they are commonly integrated as an internal component of a physical device (SUT).

The purpose of a lower test adapter is to prepare and adapt the protocol messages used by TTCN-3 test suites so that they can be transmitted successfully to the SUT. One way to achieve this goal is for example to implement lower layers and encapsulate protocol messages accordingly. For instance, CA and DEN messages need to be encapsulated in BTP datagrams, themselves encapsulated into GeoNetworking packets, and transmitted over radio link. The higher up the IUT is located in the OSI stack, the more complex is the test adapter.

TTCN-3 test suites send and receive protocol messages via TTCN-3 communication ports. For each of these ports defined in the test suites, a corresponding port entity needs to be implemented in the test. To provide maximum flexibility and allow for extensibility, the test adapter ports of the ITS test platform have been designed with the following constraints:

- For each port family, the lower stack can be configured using test adapter parameters (see annex D). As a consequence it is possible to dynamically define what will be the lower layers used to communicate with SUT, and how protocol messages will be encapsulated.
- All the instances of ports are independent.
- Behaviour of ports and lower layers can be dynamically modified by using predefined AC (Adapter Control) primitives directly sent from TTCN-3 script using dedicated port AcPort. For example, the AC primitive 'startBeaconing' requests the test adapter to start sending beacons.

The test adapter implementation mainly features `Port` and `Layer` objects. The relationship and interactions between these objects will be further detailed in clause 7.2.2, but it is important to notice the main differences between these objects, as misunderstanding their roles can lead to confusion:

- `Port` objects are the counterpart of TTCN-3 communication ports.
- `layer` and *t_layer* objects implement the minimal functionalities of a protocol layer and provide facilities for encapsulating or decapsulating packets.
- `Port` objects are configured with a lower stack composed of cascading `Layer` objects.
- For a same protocol layer, `Layer` objects usually implement more functionalities than `Port` objects.
- *params* object contains the list of the Test System parameters. The default values are overwritten by the values provided in the configuration file.

Figure 4 and Figure 5 show the sequence diagram between these classes respectively when sending and receiving a message. The port described in this example is a CAM port configured with a BTP layer/GeoNetworking layer/radio lower layer.

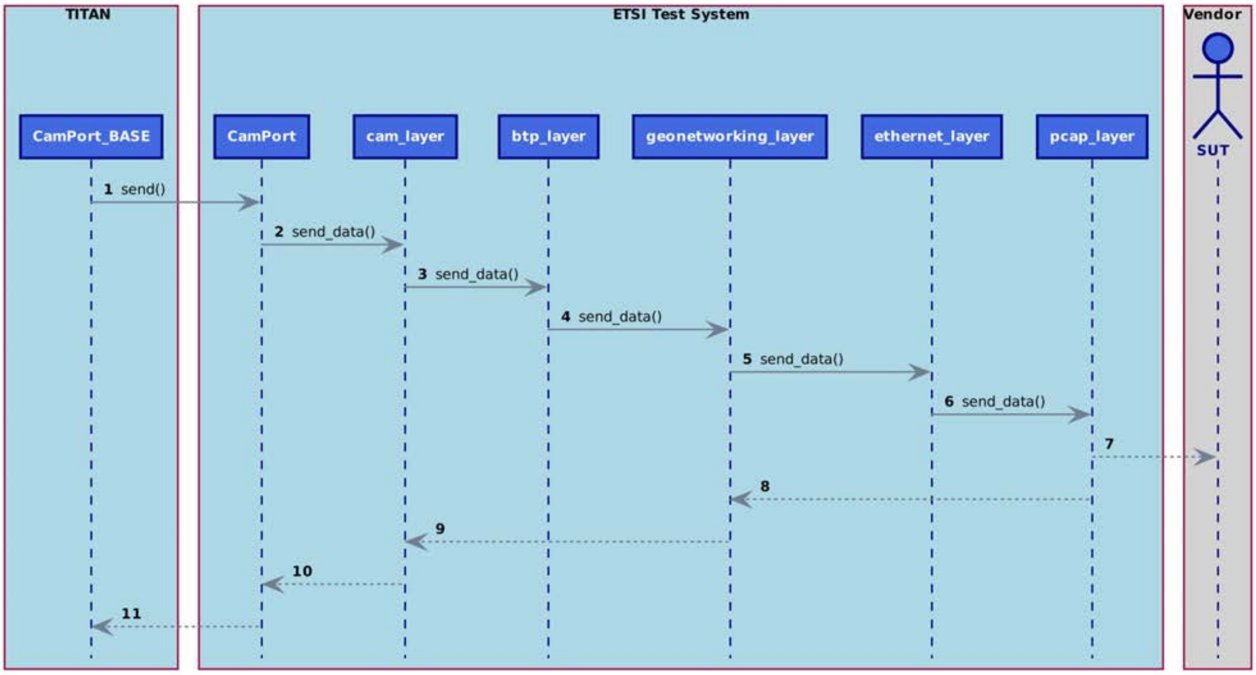


Figure 4: Sequence diagram - TTCN-s send message

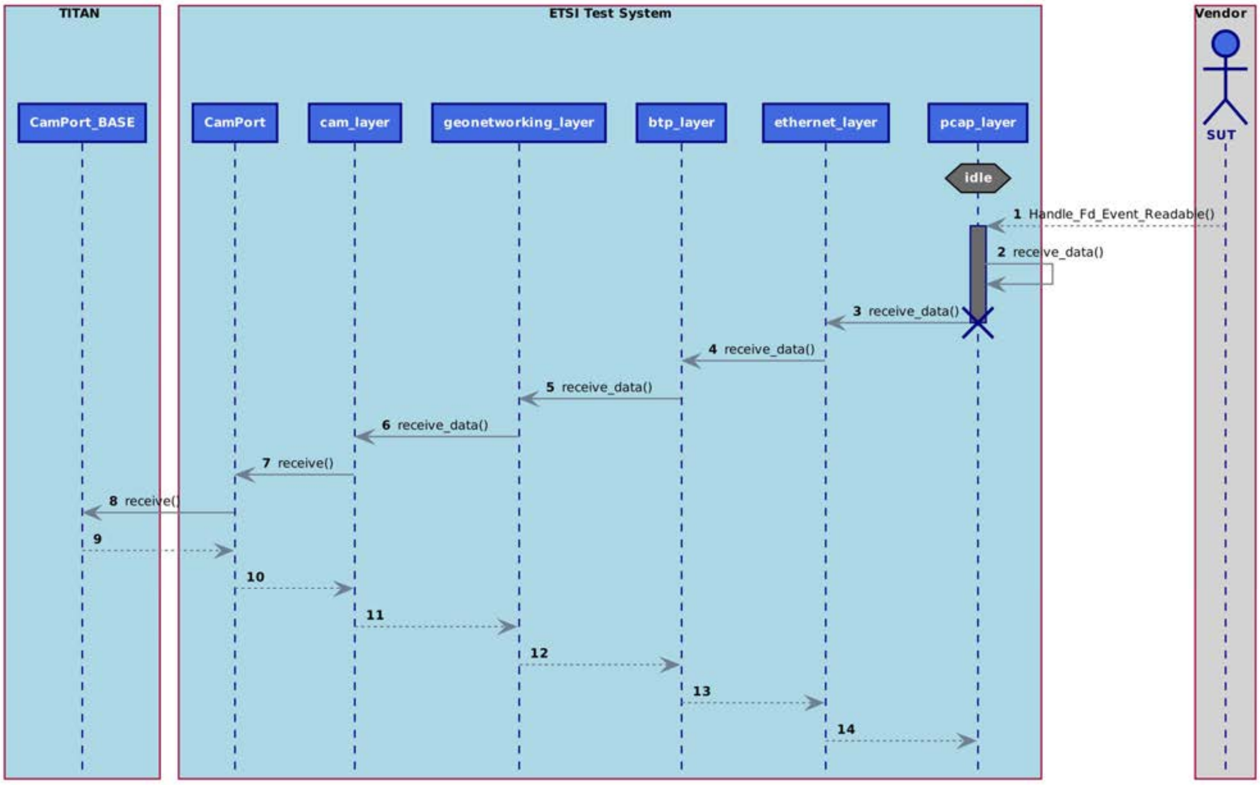


Figure 5: Sequence diagram - TTCN-s receive message

7.2.2 Advanced details of implementation

7.2.2.1 Protocol port architecture

Protocol ports realize communication between TTCN-3 and SUT.

The class diagrams below illustrate the relationship of the CamPort with the Test Execution (Figure 6) and with the Test System (Figure 7).

NOTE: The architecture is exactly the same for any TTCN-3 protocol port (e.g. DenmPort, IvimPort, etc.).

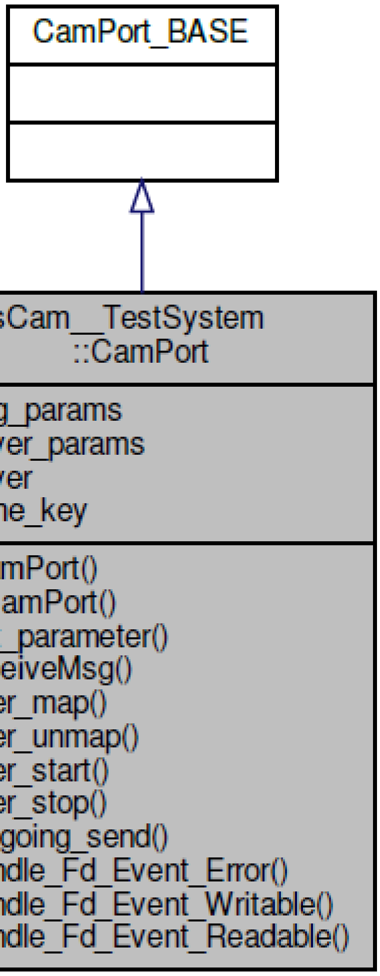


Figure 6: Class diagram - Test Execution

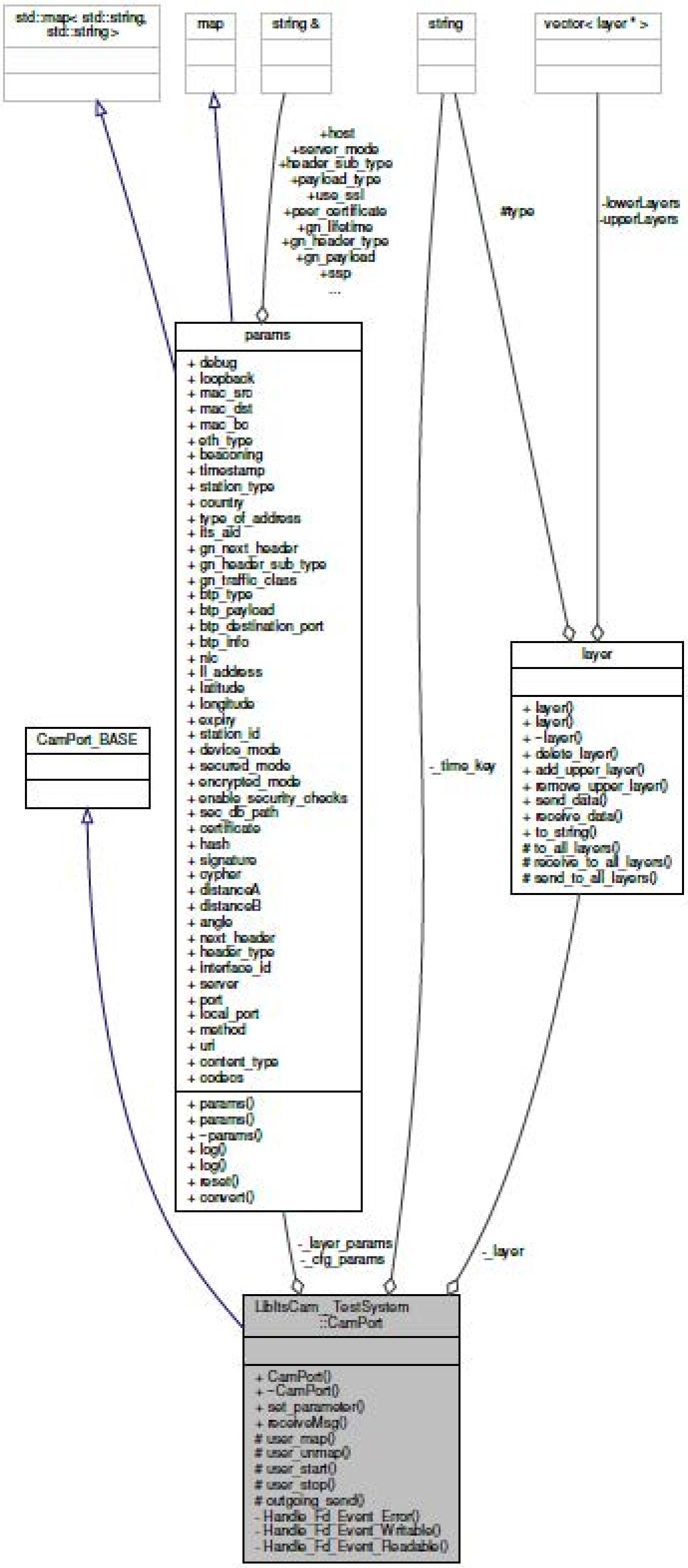


Figure 7: Class diagram - Test System

Upon protocol port initialization, lower layers are instantiated in cascade and chained as depicted in Figure 8, based on lower stack description.

Each `Layer` is responsible for encapsulating and decapsulating packets and transmitting result to lower using `send_data()`/`receive_data()` methods.

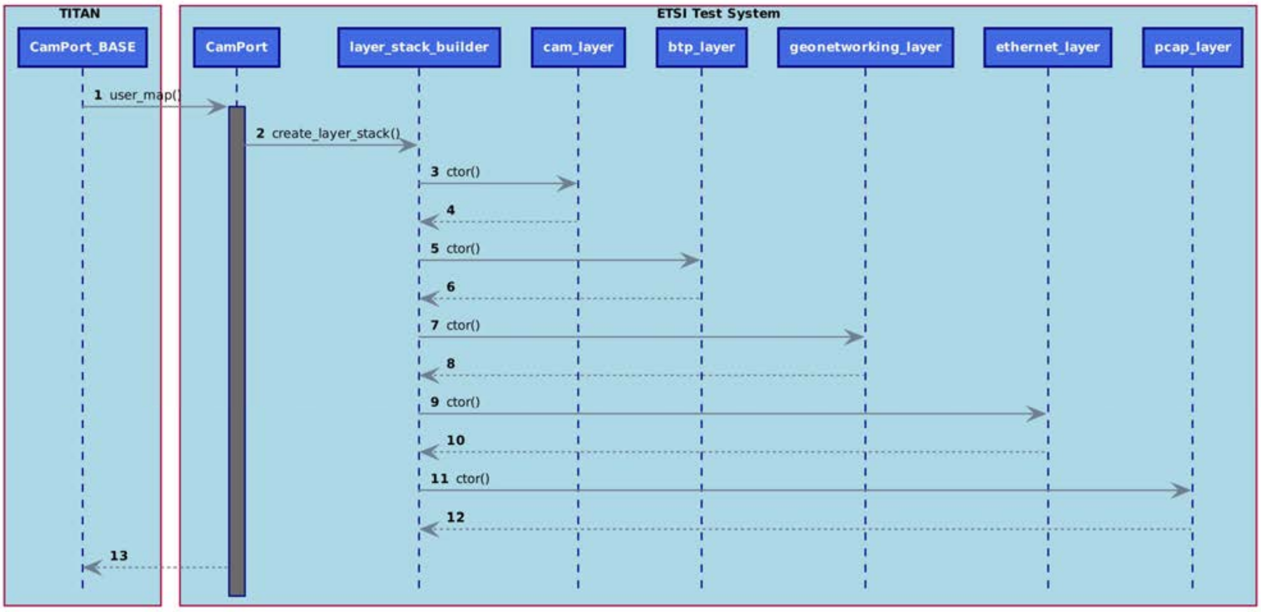


Figure 8: Port/Layer creation

Currently the following layers have been implemented:

- cam_layer: basic functionalities of GeoNetworking layer, including beaconing.
- geonetworking_layer: basic functionalities of GeoNetworking layer, including beaconing.
- btp_layer: basic functionalities of BTP layer.
- ethernet_layer. It is important to note that this class requires the usage of the external library for capturing and injecting Ethernet frames.
- Radio Access Layer.

7.2.2.2 Adapter port architecture

The adapter ports are used for the Test System configuration such as the activation of the debug mode or the activation of the security mode.

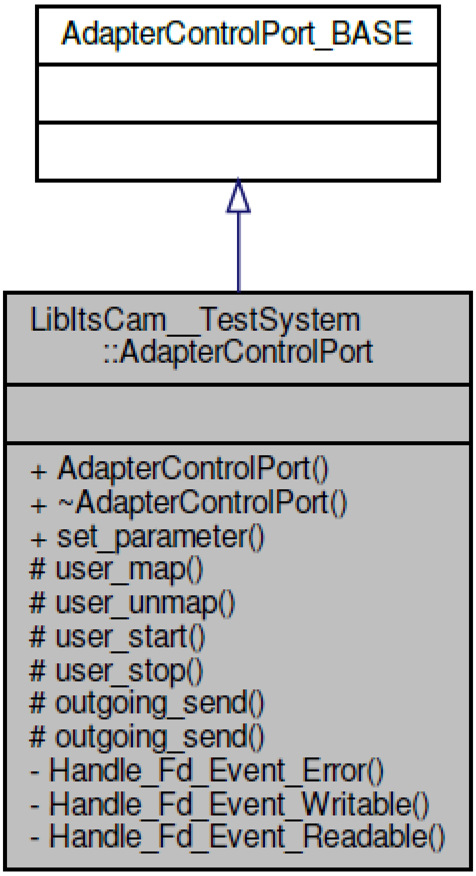


Figure 9: Adapter Control port architecture

NOTE: The architecture is exactly the same for any adapter control port (e.g. DENM, etc.).

7.2.2.3 Upper Tester port architecture

The upper tester port is used to trigger some action on the IUT such as the generation of a specific CA message.

Figure 10 shows the Upper Tester port architecture for CAM protocol.

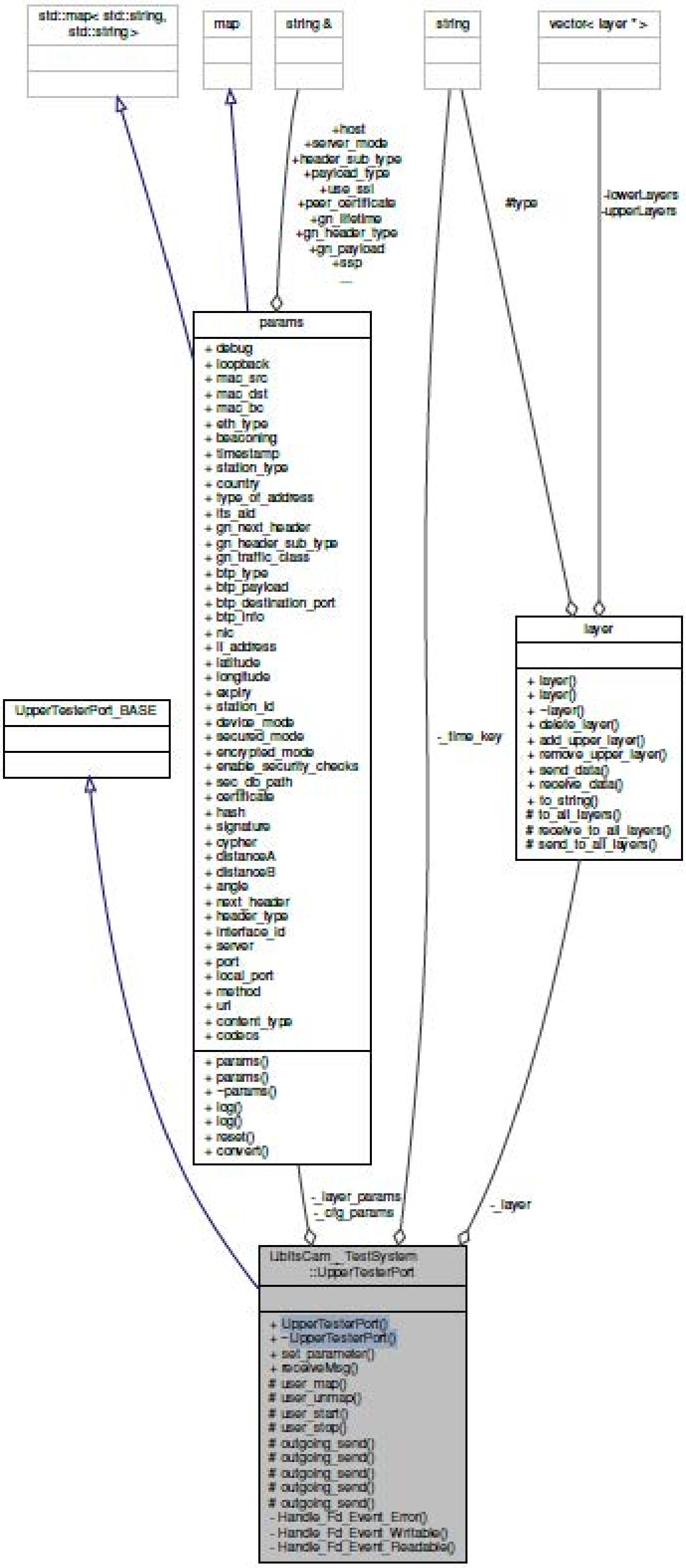


Figure 10: Class Diagram - Upper tester port

NOTE: The architecture is exactly the same for any upper tester port (e.g. DENM, etc.).

7.2.3 Extensibility of the test adapter

The test adapter can be extended in several ways. The first option is to add new protocol layers by adding new classes inheriting from the `layer` class.

It is also possible to define new protocol ports. To do so, it is necessary to implement new classes inheriting from the TITAN™ `<protocol>Port_BASE` class (e.g. IvimPort_BASE for ETSI ITS IVIM protocol). Creating a new port involve the creation of associated new layers.

7.2.4 Adapter Control primitives

The following adapter control primitives are used to control the dynamic configuration of the various layers.

Table 1: Adapter Control primitives

Adapter Control Primitive	Description
`startBeaconing`	Requests Test Adapter to start sending periodic beacons for the current component
`stopBeaconing`	Requests Test Adapter to stop sending periodic beacons for the current component
`startEnqueueingBeacons`	Requests Test Adapter to start enqueueing beacon messages on the current component GN port
`stopEnqueueingBeacons`	Requests Test Adapter to stop enqueueing beacon messages on the current component GN port
`startMultipleBeaconing`	Requests Test Adapter to start simulating neighbour presence by sending multiple periodic beacons for the current component
`stopMultipleBeaconing`	Requests Test Adapter to stop simulating neighbour presence
`getLongPositionVector`	Gets the long position vector of a neighbour given its GN_Address

7.2.5 Adapter configuration parameters

The test adapter provides several parameters to configure and adapt its behaviour. Some of those parameters are generic and apply globally to the complete test adapter, and some are specific to a particular protocol (i.e. those are mainly parameters used by Port object).

Table 2: Generic test adapter configuration parameters

Parameter name	Description	Example
`UpperTesterSettings`	IUT's Upper Tester module IP address and port, to which Test System UT primitives will be sent. <address>:<port>	`192.168.56.129:1501`
`TsLatitude`	Latitude of the Test System	`7 000`
`TsLongitude`	Longitude of the Test System	`520 000`
`LocalEthernetMAC`	Link layer address of the physical interface to be used to communicate with IUT	`005056C00008`
`IutEthernetTypeValue`	Ethertype value used by IUT	`0x8947`

Table 3: GeoNetworking test adapter configuration parameters

Parameter name	Description	Example
geoNetworkingPort	Configuration of GnPort's lower layers <layer1>/<layer2>/.../<layerN>	ETH
LinkLayer_MTC	Link layer address of simulated ITS-S MTC	BABEBABE0000
LinkLayer_NodeA	Link layer address of simulated ITS-S NodeA	BABEBABE0001
LinkLayer_NodeB	Link layer address of simulated ITS-S NodeB	BABEBABE0002
LinkLayer_NodeC	Link layer address of simulated ITS-S NodeC	BABEBABE0003
LinkLayer_NodeD	Link layer address of simulated ITS-S NodeD	BABEBABE0004
LinkLayer_NodeE	Link layer address of simulated ITS-S NodeE	BABEBABE0005
LinkLayer_NodeF	Link layer address of simulated ITS-S NodeF	BABEBABE0006
TsBeaconInterval	Beaconing interval to be used by GnPort	1 000

Table 4: BTP test adapter configuration parameters

Parameter name	Description	Example
btpPort	Configuration of BtpPort's lower layers <layer1>/<layer2>/.../<layerN>	GN/ETH

Table 5: CAM test adapter configuration parameters

Parameter name	Description	Example
camPort	Configuration of CamPort's lower layers <layer1>/<layer2>/.../<layerN>	BTP/GN/ETH

Table 6: DENM test adapter configuration parameters

Parameter name	Description	Example
denmPort	Configuration of DenmPort's lower layers <layer1>/<layer2>/.../<layerN>	BTP/GN/ETH

Table 7: GN6 test adapter configuration parameters

Parameter name	Description	Example
ipv6OverGeoNetworkingPort	Configuration of Gn6Port's lower layers <layer1>/<layer2>/.../<layerN>	Debug
Gn6RemoteAdapterIp	IP address of the GN6 remote adapter	192.168.56.11
Gn6RemoteAdapterPort	Listening port of the remote GN6 adapter	42 000

Table 8: Security test adapter configuration parameters

Parameter	Description	Example
TsSecuredPath	Secured root path to access certificate files	"data/certificates"
TsSecuredConfiId	Vendor specific configuration identifier. This should be actually a name of the subfolder inside the TsSecuredPath, containing the IUT certificates or digests, e.g. "data/certificates/vendorA"	vendorA
NOTE 1: The parameter TsSecuredMode==true indicates that all security tasks are performed by the test adapter. This includes that the test adapter will decapsulate the received secured message and pass the payload to the upper layer as well as to encapsulate the toBeSent message. The parameter TsSecuredMode==false indicates that the test adapter passes the received secured message to the upper layer. The test adapter does not perform any security tasks on the toBeSentMessage. NOTE 2: There are three possible ways of executing the tests: - Running CAM/DENM/GN tests with IUT in secured mode: TsSecuredMode set to TRUE and PICS_GN_SECURITY set to FALSE - Running CAM/DENM/GN tests with IUT in non-secured mode: TsSecuredMode set to FALSE and PICS_GN_SECURITY set to FALSE - Running Security tests with IUT in secured mode: TsSecuredMode set to FALSE and PICS_GN_SECURITY set to TRUE		

7.3 Platform Adapter

All TTCN-3 commercial tools provide generic Platform Adapter implementations for managing TTCN-3 timers. These implementations are well tested and usually accurate enough for most usages. In the case of ITS protocols, e.g. DENM re-broadcasting, GN beacon interval, etc., the protocol timer value is in the order of magnitude of hundreds of milliseconds. This order of magnitude can be handled well with the built in test system timers. As a consequence, no specific development is required for this component.

7.4 Upper Tester

The upper tester is used to interact with the upper interface of the implementation under test (IUT). It is typically implemented as an upper tester module executing in the test adapter and as a small module executing on the SUT and acts as an upper layer for the IUT, as shown in Figure 11. It is particularly useful for:

- Triggering events in SUT.
- Triggering messages.
- Checking that payload are transmitted correctly to upper layers.

The communication between the two upper tester modules is performed in accordance with the upper tester message format described in annex C.

As it interacts with potentially proprietary APIs, it is usually the responsibility of IUT vendors to implement module located within SUT.

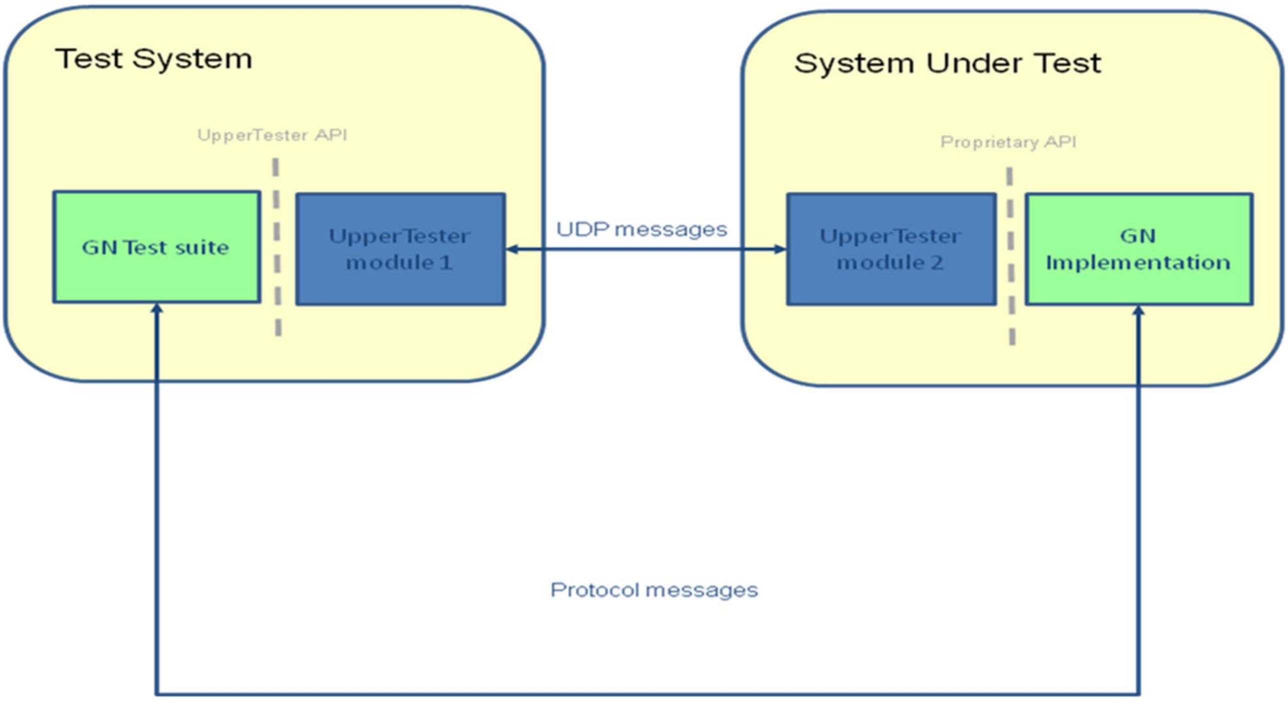


Figure 11: Upper Tester architecture

This upper tester module implements the upper tester message based API described in annex C.

Annex A:
Codecs Source Code

The reference implementation of the codec source code can be found on the ETSI forge repository: https://forge.etsi.org/rep/ITS/TS.ITS.git.

Annex B:
Test Adapter Source Code

The reference implementation of the test adapter source code can be found on the ETSI forge repository: https://forge.etsi.org/rep/ITS/TS.ITS.git.

Annex C: Upper Tester Message Format

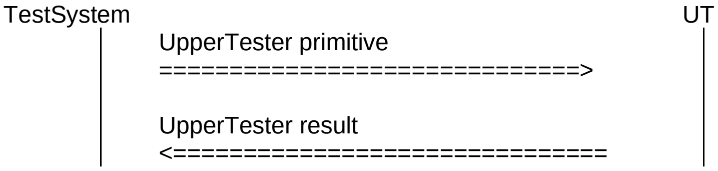
C.1 Introduction

The messages defined in the present annex are exchanged between the Test System and Upper Tester using a UDP connection.

All integer values are encoded in big-endian byte order (most significant byte first).

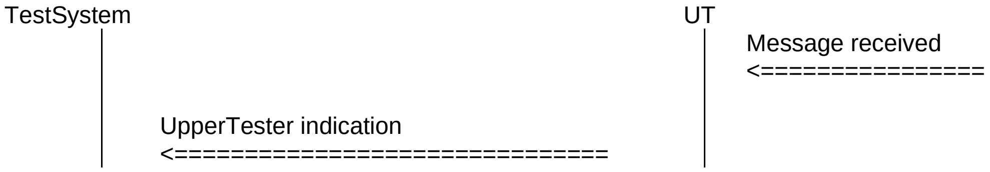
Two different message exchanges can occur:

- The first communication exchange is initiated by the test system and consists in a request - response exchange as described below. The UpperTester result message is specific to each primitive and may be used to indicate the success of the request or to report some values.



In this case the UDP destination port of the response is identical to the UDP source port of the corresponding request. When receiving UtInitialize primitive from Test System, the UDP source port of this request is saved as 'defaultUTPort' and used for unsolicited indications.

- The second communication exchange is initiated by the Upper Tester. It consists in unsolicited indications sent each time a packet is transmitted to upper layers, as described below. The Test System never replies to such messages (one way communication).



In this case, the UDP destination port of the indication is set to the 'defaultUTPort', which corresponds to the UDP source port of the UTInitialize request.

Format of UtResult:

<table>
<tr><td>0</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td colspan="8">MessageType = 0x24</td><td colspan="8">Result</td></tr>
</table>

Name	Length	Value
MessageType	1 byte	0x24
Result	1 byte	0x00: Failure 0x01: Success

C.2 Common Upper Tester Primitives

C.2.1 UtInitialize

NOTE: The notation "TS ➔ UT" and "UT ➔ TS" is used in this clause and all subsequent clauses, and signifies "from TS to UT" and "from UT to TS".

This message is used to request initialization of IUT implementation. This means that at least:

- location table, forwarding buffers, LS buffer, list of collected certificates should be cleared; and
- the Sequence Number and the GN address should be reset to initially configured values.

Request (UtInitialize TS ➔ UT):

0	
0 1 2 3 4 5 6 7	
MessageType = 0x00	HashedId8
...	
...	

Name	Length	Value
MessageType	1 byte	0x00
HashedId8	8 bytes	In case PICS_GN_SECURITY is set to TRUE, then HashedId8 indicates the AT certificate digest to be used by the IUT. In case PICS_GN_SECURITY is set to FALSE, then HashedId8 is set to 0, which indicates to the IUT that testing of the security protocol is disabled.

Response (UtInitializeResult UT ➔ TS):

0	1
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0x01	Result

Name	Length	Value
MessageType	1 byte	0x01
Result	1 byte	0x00: Failure 0x01: Success

C.2.2 ChangePosition

This message is used to change the position of the ITS station. The latitude, longitude and altitude parameters are relative to the current position of IUT. They are NOT absolute position.

Request (UtChangePosition TS ➔ UT):

0	1	2	3
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0x02	DeltaLatitude		
...	DeltaLongitude		
...	DeltaAltitude		
...			

Name	Length	Value
MessageType	1 byte	0x02
DeltaLatitude	4 bytes	Latitude offset (multiples of 0,1 microdegree)
DeltaLongitude	4 bytes	Longitude offset (multiples of 0,1 microdegree)
DeltaElevation	4 bytes	Altitude offset (centimetre)

Response (UtChangePositionResult UT ➔ TS):

```
0                       1
0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7
|MessageType = 0x03     |Result                 |
```

Name	Length	Value
MessageType	1 byte	0x03
Result	1 byte	0x00: Failure 0x01: Success

C.2.3 ChangePseudonym

This message is used to change the pseudonym of the ITS-S.

Request (UtChangePseudonym TS ➔ UT):

```
0
0  1  2  3  4  5  6  7
|MessageType = 0x04     |
```

Name	Length	Value
MessageType	1 byte	0x04

Response (UtChangePseudonymResult UT ➔ TS):

```
0                       1
0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7
|MessageType = 0x05     |Result                 |
```

Name	Length	Value
MessageType	1 byte	0x05
Result	1 byte	0x00: Failure 0x01: Success

C.3 CAM Upper Tester Primitives

C.3.1 ChangeCurvature

This message is used to set the curvature of the ITS station. The curvature parameter is relative to the current curvature value. It is NOT an absolute value.

Request (UtCamTrigger_changeCurvature TS ➔ UT):

```
0                       1                       2
0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7
|MessageType = 0x30     |Curvature                                      |
```

Name	Length	Value
MessageType	1 byte	0x30
Curvature	2 bytes	Signed integer. Curvature offset from -30 000 to 30 001

Response (UtCamTriggerResult UT ➔ TS):

<table>
<tr><td colspan="8">0</td><td colspan="8">1</td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td colspan="8">MessageType = 0x21</td><td colspan="8">Result</td></tr>
</table>

Name	Length	Value
MessageType	1 byte	0x21
Result	1 byte	0x00: Failure 0x01: Success

C.3.2 ChangeSpeed

This message is used to change the speed of the ITS station. The vehicle speed is increased by the value of 'SpeedVariation' field.

For instance, if the current speed of the ITS station is 10 m/s and received SpeedVariation is +300, then the new vehicle speed will be 10 + 0,01 × 300 = 13 m/s.

Request (UtCamTrigger_changeSpeed TS ➔ UT):

<table>
<tr><td colspan="8">0</td><td colspan="8">1</td><td colspan="8">2</td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td colspan="8">MessageType = 0x31</td><td colspan="16">SpeedVariation</td></tr>
</table>

Name	Length	Value
MessageType	1 byte	0x31
SpeedVariation	2 bytes	Signed integer. Speed variation in units of cm/s

Response (UtCamTriggerResult UT ➔ TS):

<table>
<tr><td colspan="8">0</td><td colspan="8">1</td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td colspan="8">MessageType = 0x21</td><td colspan="8">Result</td></tr>
</table>

Name	Length	Value
MessageType	1 byte	0x21
Result	1 byte	0x00: Failure 0x01: Success

C.3.3 SetAccelerationControlStatus

This message is used to set acceleration control status of the ITS station.

Request (UtCamTrigger_setAccelerationControlStatus TS ➔ UT):

<table>
<tr><td colspan="8">0</td><td colspan="8">1</td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td colspan="8">MessageType = 0x32</td><td>B</td><td>G</td><td>E</td><td>C</td><td>A</td><td>CC</td><td>L</td><td>X</td></tr>
</table>

Name	Length	Value
MessageType	1 byte	0x32
B	1 bit	0: brake pedal inactive 1: brake pedal active
G	1 bit	0: gas pedal inactive 1: gas pedal active
E	1 bit	0: emergency brake inactive 1: emergency brake active
C	1 bit	0: collision warning inactive 1: collision warning active
A	1 bit	0: ACC inactive 1: ACC active
CC	1 bit	0: cruise control inactive 1: cruise control active
L	1 bit	0: speed limiter inactive 1: speed limiter active
X	1 bit	Reserved

Response (UtCamTriggerResult UT ➔ TS):

0								1							
0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
MessageType = 0x21								Result							

Name	Length	Value
MessageType	1 byte	0x21
Result	1 byte	0x00: Failure 0x01: Success

C.3.4 SetExteriorLightsStatus

This message is used to set exterior lights status of the ITS station.

Request (UtCamTrigger_setExteriorLightsStatus TS ➔ UT):

0								1							
0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
MessageType = 0x33								LB	HB	LT	RT	D	R	F	P

Name	Length	Value
MessageType	1 byte	0x33
LB	1 bit	0: low beam headlights off 1: low beam headlights on
HB	1 bit	0: high beam off 1: high beam headlights on
LT	1 bit	0: left turn signal off 1: left turn signal on
RT	1 bit	0: right turn signal off 1: right turn signal on
D	1 bit	0: daytime running lights off 1: daytime running lights on
R	1 bit	0: reverse light off 1: reverse lights on
F	1 bit	0: fog light off 1: fog light on
P	1 bit	0: parking lights off 1: parking lights on

Response (UtCamTriggerResult UT ➔ TS):

<table>
<tr><td colspan="8">0</td><td colspan="8">1</td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td colspan="8">MessageType = 0x21</td><td colspan="8">Result</td></tr>
</table>

Name	Length	Value
MessageType	1 byte	0x32
Result	1 byte	0x00: Failure 0x01: Success

C.3.5 ChangeHeading

This message is used to change the heading of the ITS station. The heading parameter is relative to the current heading value. It is NOT an absolute value.

Request (UtCamTrigger_changeHeading TS ➔ UT):

<table>
<tr><td colspan="8">0</td><td colspan="8">1</td><td colspan="8">2</td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td colspan="8">MessageType = 0x34</td><td colspan="16">Heading</td></tr>
</table>

Name	Length	Value
MessageType	1 byte	0x34
Heading	2 bytes	Heading offset. Integer value from 0 to 3 600

Response (UtCamTriggerResult UT ➔ TS):

<table>
<tr><td colspan="8">0</td><td colspan="8">1</td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td colspan="8">MessageType = 0x21</td><td colspan="8">Result</td></tr>
</table>

Name	Length	Value
MessageType	1 byte	0x21
Result	1 byte	0x00: Failure 0x01: Success

C.3.6 SetDriveDirection

This message is used to change the direction of the ITS station.

Request (UtCamTrigger_setDriveDirection TS ➔ UT):

<table>
<tr><td colspan="8">0</td><td colspan="8">1</td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td colspan="8">MessageType = 0x35</td><td colspan="8">Direction</td></tr>
</table>

Name	Length	Value
MessageType	1 byte	0x35
Direction	1 byte	0x00: Forward 0x01: Backward 0x02: Unavailable

Response (UtCamTriggerResult UT ➔ TS):

```
0                               1
0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7
| MessageType = 0x21          | Result                        |
```

Name	Length	Value
MessageType	1 byte	0x21
Result	1 byte	0x00: Failure 0x01: Success

C.3.7 ChangeYawRate

This message is used to change the yaw rate of the ITS station. The yaw rate parameter is relative to the current yaw rate value. It is NOT an absolute value.

Request (UtCamTrigger_changeYawRate TS ➔ UT):

```
0                               1                               2
0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7
| MessageType = 0x36          | YawRate                                                       |
```

Name	Length	Value
MessageType	1 byte	0x36
YawRate	2 bytes	Yaw rate offset. Signed integer from -32 766 to 32 767

Response (UtCamTriggerResult UT ➔ TS):

```
0                               1
0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7
| MessageType = 0x21          | Result                        |
```

Name	Length	Value
MessageType	1 byte	0x21
Result	1 byte	0x00: Failure 0x01: Success

C.3.8 CamEventIndication

This message is used to indicate reception of CAM information by IUT.

Indication (UtCamEventInd UT ➔ TS):

```
0                               1                               2                               3
0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7
| MessageType = 0x23          | CamPduLength                                                  | CamPdu                        :
:                                                  ...                                                                    |
```

Name	Length	Value
MessageType	1 byte	0x23
CamPduLength	2 bytes	Length of 'CamPdu' field
CamPdu	Variable	Received CAM

C.3.9 SetStationType

This message is used to change the type of the ITS station.

Request (UtCamTrigger_setStationType TS ➔ UT):

0								1								2
0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	
MessageType = 0x39								StationType								

Name	Length	Value
MessageType	1 byte	0x39
StationType	1 byte	Unsigned char range from 0 to 15 unknown(0), pedestrian(1), cyclist(2), moped(3), motorcycle(4), passengerCar(5), bus(6), lightTruck(7), heavyTruck(8), trailer(9), specialVehicles(10), tram(11), roadSideUnit(15)

Response (UtCamTriggerResult UT ➔ TS):

0								1							
0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
MessageType = 0x21								Result							

Name	Length	Value
MessageType	1 byte	0x21
Result	1 byte	0x00: Failure 0x01: Success

C.3.10 SetVehicleRole

This message is used to change the vehicle role of the ITS station.

Request (UtCamTrigger_setVehicleRole TS ➔ UT):

0								1								2
0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	
MessageType = 0x3a								VehiculeRole								

Name	Length	Value
MessageType	1 byte	0x3a
VehiculeRole	1 byte	Unsigned char range from 0 to 7 default(0), publicTransport(1), specialTransport(2), dangerousGoods(3), roadWork(4), rescue(5), emergency(6), safetyCar(7)

Response (UtCamTriggerResult UT ➔ TS):

0	1
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0x21	Result

Name	Length	Value
MessageType	1 byte	0x21
Result	1 byte	0x00: Failure 0x01: Success

C.3.11 SetEmbarkationStatus

This message is used to indicate whether the passenger embarkation is ongoing.

Request (UtCamTrigger_setEmbarkationStatus TS ➔ UT):

0	1	2
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	
MessageType = 0x3b	EmbarkationStatus	

Name	Length	Value
MessageType	1 byte	0x3b
EmbarkationStatus	1 byte	Unsigned char. Value is 0 for false and value is 255 for true

Response (UtCamTriggerResult UT ➔ TS):

0	1
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0x21	Result

Name	Length	Value
MessageType	1 byte	0x21
Result	1 byte	0x00: Failure 0x01: Success

C.3.12 SetPtActivation

This message is used to control traffic lights, barriers, etc.

Request (UtCamTrigger_setPtActivation TS ➔ UT):

0	1	2	
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	
MessageType = 0x3c	PtActivationType	PtActivationDataLength	PtActivatioData
	...		

Name	Length	Value
MessageType	1 byte	0x3c
PtActivationType	1 byte	Unsigned char range from 0 to 255
PtActivationDataLength	1 byte	Unsigned char range from 0 to 20
PtActivatioData	Variable	Unsigned char range from 0 bytes to 20 bytes

Response (UtCamTriggerResult UT ➔ TS):

```
0                              1
0  1   2   3   4   5   6   7   0   1   2   3   4   5   6   7
|MessageType = 0x21           |Result                        |
```

Name	Length	Value
MessageType	1 byte	0x21
Result	1 byte	0x00: Failure 0x01: Success

C.3.13 SetDangerousGoods

This message is used to set the dangerous good property of the ITS station.

Request (UtCamTrigger_setDangerousGoods TS ➔ UT):

```
0                              1                              2
0  1   2   3   4   5   6   7   0   1   2   3   4   5   6   7
|MessageType = 0x3d           |DangerousGood               |
```

Name	Length	Value
MessageType	1 byte	0x3d
DangerousGood	1 byte	Unsigned char range from 0 to 19 explosives1(0), explosives2(1), explosives3(2), explosives4(3), explosives5(4), explosives6(5), flammableGases(6), nonFlammableGases(7), toxicGases(8), flammableLiquids(9), flammableSolids(10), substancesLiableToSpontaneousCombustion(11), substancesEmittingFlammableGasesUponContactWithWater(12), oxidizingSubstances(13), organicPeroxides(14), toxicSubstances(15), infectiousSubstances(16), radioactiveMaterial(17), corrosiveSubstances(18), miscellaneousDangerousSubstances(19)

Response (UtCamTriggerResult UT ➔ TS):

```
0                              1
0  1   2   3   4   5   6   7   0   1   2   3   4   5   6   7
|MessageType = 0x21           |Result                        |
```

Name	Length	Value
MessageType	1 byte	0x21
Result	1 byte	0x00: Failure 0x01: Success

C.3.14 SetLightBarSiren

This message is used to set light and siren bar status of the ITS station.

Request (UtCamTrigger_setLightBarSiren TS ➔ UT):

0								1							
0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
MessageType = 0x3f								LB	S						

Name	Length	Value
MessageType	1 byte	0x3f
LB	1 bit	0: Light bar is not activated 1: Light bar is activated
S	1 bit	0: Siren is off 1: Siren is on

Response (UtCamTriggerResult UT ➔ TS):

0								1							
0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
MessageType = 0x21								Result							

C.4 DENM Upper Tester Primitives

C.4.1 GenerateDenmEvent

This message is used to create a new DENM event.

Request (UtDenmTrigger TS ➔ UT):

0								1								2								3							
0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
MessageType = 0x10								V	R	X	T	X	K	I	X	DetectionTime															
…																															
ValidityDuration																								RepetitionDuration							
…																InfoQuality								Cause							
SubCause								RelevanceDistance								RelevanceTrafficDirection								TransmissionInterval							
								RepetitionInterval																alacarteLength							
alacarte																															

Name	Length	Value
MessageType	1 byte	0x10
V	1 bit	0: ValidityDuration to be ignored 1: ValidityDuration to be used
R	1 bit	0: RepetitionDuration to be ignored 1: RepetitionDuration to be used
X	1 bit	reserved
T	1 bit	0: RelevanceTrafficDirection to be ignored 1: RelevanceTrafficDirection to be used
X	1 bit	reserved
K	1 bit	0: TransmissionInterval to be ignored 1: TransmissionInterval to be used
I	1 bit	0: RepetitionInterval to be ignored 1: RepetitionInterval to be used
X	1 bit	reserved
DetectionTime	6 bytes	Unsigned integer. From 0 to 3 153 600 000 000
ValidityDuration	3 bytes	Unsigned integer. From 0 s to 86 400 s
RepetitionDuration	3 bytes	Unsigned integer. From 0 s to 86 400 s
InfoQuality	1 byte	0x00: Unavailable 0x01: Lowest … 0x07: Highest
Cause	1 byte	Event cause ID
Subcause	1 byte	Event sub-cause ID
RelevanceDistance	1 byte	0x00: less than 50 m 0x01: less than 100 m 0x02: less than 200 m 0x03: less than 500 m 0x04: less than 1 000 m 0x05: less than 5 km 0x06: less than 10 km 0x07: greater than 10 km
RelevanceTrafficDirection	1 byte	0x00: unavailable 0x01: upstream traffic 0x02: downstream traffic 0x03: all traffic directions
TransmissionInterval	2 bytes	From 1 ms to 10 000 ms
RepetitionInterval	2 bytes	From 1 ms to 10 000 ms
alacarteLength	1 byte	Length of 'Alacarte container' field Value 0 means no Alacarte container included
alacarte	n bytes	Alacarte container

Response (UtDenmTriggerResult UT ➔ TS):

<table>
<tr><td colspan="8">0</td><td colspan="8">1</td><td colspan="8">2</td><td colspan="8">3</td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td colspan="8">MessageType = 0x11</td><td colspan="8">Result</td><td colspan="16">StationId</td></tr>
<tr><td colspan="16">...</td><td colspan="16">SequenceNo</td></tr>
</table>

Name	Length	Value
MessageType	1 byte	0x11
Result	1 byte	Operation result
StationId	4 bytes	Station ID
SequenceNo	2 bytes	Event sequence number

C.4.2 UpdateDenmEvent

This message is used to update expiration time of an existing DENM event.

Request (UtDenmUpdate TS ➔ UT):

0	1								2	3
0 1 2 3 4 5 6 7	0	1	2	3	4	5	6	7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0x12	V	S	D	T	C	K	I	X	StationId	
…									SequenceNo	
DetectionTime										
…									ValidityDuration	
…	InfoQuality								Cause	SubCause
RelevanceDistance	RelevanceTrafficDirection								TransmissionInterval	
RepetitionInterval									alacarteLength	alacarte

Name	Length	Value
MessageType	1 byte	0x12
V	1 bit	0: ValidityDuration to be ignored 1: ValidityDuration to be used
S	1 bit	0: InfoQuality, CauseCode and SubCauseCode to be ignored 1: InfoQuality, CauseCode and SubCauseCode to be used
D	1 bit	0: RelevanceDistance to be ignored 1: RelevanceDistance to be used
T	1 bit	0: RelevanceTrafficDirection to be ignored 1: RelevanceTrafficDirection to be used
C	1 bit	0: TrafficClass to be ignored 1: TrafficClass to be used
K	1 bit	0: TransmissionInterval to be ignored 1: TransmissionInterval to be used
I	1 bit	0: RepetitionInterval to be ignored 1: RepetitionInterval to be used
X	1 bit	reserved
StationId	4 bytes	Original event's station ID
SequenceNo	2 bytes	Original event's sequence number
DetectionTime	6 bytes	Unsigned integer. From 0 to 3 153 600 000 000
ValidityDuration	3 bytes	Unsigned integer. From 0 s to 86 400 s
InfoQuality	1 byte	0x00: Unavailable 0x01: Lowest … 0x07: Highest
Cause	1 byte	Event cause ID
Subcause	1 byte	Event sub-cause ID
RelevanceDistance	1 byte	0x00: less than 50 m 0x01: less than 100 m 0x02: less than 200 m 0x03: less than 500 m 0x04: less than 1 000 m 0x05: less than 5 km 0x06: less than 10 km 0x07: greater than 10 km
RelevanceTrafficDirection	1 byte	0x00: all traffic directions 0x01: upstream traffic 0x02: downstream traffic 0x03: opposite traffic
TransmissionInterval	2 bytes	From 1 ms to 10 000 ms
RepetitionInterval	2 bytes	From 1 ms to 10 000 ms
alacarteLength	1 byte	Length of 'Alacarte container' field Value 0 means no Alacarte container included
alacarte	n bytes	Alacarte container

Response (UtDenmUpdateResult UT ➔ TS):

```
0                       1                       2                       3
0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7
|MessageType = 0x13    |Result                 |StationId                                     :
|...                                           |SequenceNo                                    |
```

Name	Length	Value
MessageType	1 byte	0x13
Result	1 byte	Operation result
StationId	4 bytes	Station ID
SequenceNo	2 bytes	Event sequence number

C.4.3 TerminateDenmEvent

This message is used to terminate an existing DENM event.

Request (UtDenmTermination TS ➔ UT):

```
0                       1                       2                       3
0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7
|MessageType = 0x14    |StationId                                                             :
:...                   |SequenceNo                                      |
```

Name	Length	Value
MessageType	1 byte	0x14
StationId	4 bytes	Original Station ID
SequenceNo	2 bytes	Event sequence number

Response (UtDenmTerminationResult UT ➔ TS):

```
0                       1                       2                       3
0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7
|MessageType = 0x15    |Result                 |
```

Name	Length	Value
MessageType	1 byte	0x15
Result	1 byte	0x00: Failure 0x01: Success

C.4.4 DenmEventIndication

This message is used to indicate reception of DENM information by IUT.

Indication (UtDenmEventInd UT ➔ TS):

```
0                       1                       2                       3
0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7
|MessageType = 0x17    |DenmPduLength                                  |DenmPdu               :
:                                  ...                                                        |
```

Name	Length	Value
MessageType	1 byte	0x17
DenmPduLength	2 bytes	Length of 'DenmPdu' field
DenmPdu	Variable	Received DENM

C.5 GeoNetworking Upper Tester Primitives

C.5.1 GenerateGeoUnicast

This message is used to trigger a GeoUnicast message.

Request (UtGnTrigger_geoUnicast TS ➔ UT):

0	1	2	3
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0x50	DstGnAddress		
...			
...	Lifetime		TrafficClass
PayloadLength		Payload	

Name	Length	Value
MessageType	1 byte	0x50
DstGnAddr	8 bytes	Destination GN Address
Lifetime	2 bytes	Packet lifetime in milliseconds
TrafficClass	1 byte	Packet traffic class
PayloadLength	2 bytes	Length of 'Payload' field
Payload	Variable	Packet's final payload

Response (UtGnTriggerResult UT ➔ TS):

0	1
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0x41	Result

Name	Length	Value
MessageType	1 byte	0x41
Result	1 byte	0x00: Failure 0x01: Success

C.5.2 GenerateGeoBroadcast

This message is used to trigger a GeoBroadcast message.

Request (UtGnTrigger_geoBroadcast TS ➔ UT):

0	1	2	3
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0x51	Shape	Lifetime	
TrafficClass	Reserved		
Latitude			
Longitude			
DistanceA		DistanceB	
Angle		PayloadLength	
Payload			

Name	Length	Value
MessageType	1 byte	0x51
Shape	1 byte	0: Circle 1: Rectangle 2: Ellipse
Lifetime	2 bytes	Packet lifetime in milliseconds
TrafficClass	1 byte	Packet traffic class
Reserved	3 bytes	Reserved
Latitude	4 bytes	Destination area latitude (1/10 degrees)
Longitude	4 bytes	Destination area longitude (1/10 degrees)
DistanceA	2 bytes	Destination area distance A
DistanceB	2 bytes	Destination area distance B
Angle	2 bytes	Destination area angle
PayloadLength	2 bytes	Length of 'Payload' field
Payload	Variable	Packet's final payload

Response (UtGnTriggerResult UT ➔ TS):

<table>
<tr><td colspan="8">0</td><td colspan="8">1</td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td colspan="8">MessageType = 0x41</td><td colspan="8">Result</td></tr>
</table>

Name	Length	Value
MessageType	1 byte	0x41
Result	1 byte	0x00: Failure 0x01: Success

C.5.3 GenerateGeoAnycast

This message is used to trigger a GeoAnycast message.

Request (UtGnTrigger_geoAnycast TS ➔ UT):

<table>
<tr><td colspan="8">0</td><td colspan="8">1</td><td colspan="8">2</td><td colspan="8">3</td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td colspan="8">MessageType = 0x52</td><td colspan="8">Shape</td><td colspan="16">Lifetime</td></tr>
<tr><td colspan="8">TrafficClass</td><td colspan="24">Reserved</td></tr>
<tr><td colspan="32">Latitude</td></tr>
<tr><td colspan="32">Longitude</td></tr>
<tr><td colspan="16">DistanceA</td><td colspan="16">DistanceB</td></tr>
<tr><td colspan="16">Angle</td><td colspan="16">PayloadLength</td></tr>
<tr><td colspan="32">Payload</td></tr>
</table>

Name	Length	Value
MessageType	1 byte	0x52
Shape	1 byte	0: Circle 1: Rectangle 2: Ellipse
Lifetime	2 bytes	Packet lifetime in milliseconds
TrafficClass	1 byte	Packet traffic class
Reserved	3 bytes	Reserved
Latitude	4 bytes	Destination area latitude (1/10 degrees)
Longitude	4 bytes	Destination area longitude (1/10 degrees)
DistanceA	2 bytes	Destination area distance A
DistanceB	2 bytes	Destination area distance B
Angle	2 bytes	Destination area angle
PayloadLength	2 bytes	Length of 'Payload' field
Payload	Variable	Packet's final payload

Response (UtGnTriggerResult UT ➔ TS):

0	1
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0x41	Result

Name	Length	Value
MessageType	1 byte	0x41
Result	1 byte	0x00: Failure 0x01: Success

C.5.4 GenerateSHB

This message is used to trigger a SHB message.

Request (UtGnTrigger_shb TS ➔ UT):

0	1	2	3
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0x53	TrafficClass	PayloadLength	
Payload			

Name	Length	Value
MessageType	1 byte	0x53
TrafficClass	1 byte	Packet traffic class
PayloadLength	2 bytes	Length of 'Payload' field
Payload	Variable	Packet's final payload

Response (UtGnTriggerResult UT ➔ TS):

0	1
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0x41	Result

Name	Length	Value
MessageType	1 byte	0x41
Result	1 byte	0x00: Failure 0x01: Success

C.5.5 GenerateTSB

This message is used to trigger a TSB message.

Request (UtGnTrigger_tsb TS ➔ UT):

0	1	2	3
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0x54	NbHops	Lifetime	
TrafficClass	PayloadLength		Payload

Name	Length	Value
MessageType	1 byte	0x54
NbHops	1 byte	Number of hops
Lifetime	2 bytes	Packet lifetime in milliseconds
TrafficClass	1 byte	Packet traffic class
PayloadLength	2 bytes	Length of 'Payload' field
Payload	Variable	Packet's final payload

Response (UtGnTriggerResult UT ➔ TS):

0	1
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0x41	Result

Name	Length	Value
MessageType	1 byte	0x41
Result	1 byte	0x00: Failure 0x01: Success

C.5.6 GnEventIndication

This message is used to check whether payload contained in GeoNetworking PDU has been transmitted to upper layer (CAM/DENM/IPv6).

Indication (UtGnEventInd UT ➔ TS):

0	1	2	3
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0x55	PacketLength		Packet
...			

Name	Length	Value
MessageType	1 byte	0x17
PacketLength	2 bytes	Length of 'Packet' field
Packet	Variable	Packet's final payload

C.6 IPv6OverGeoNetworking Upper Tester Primitives

C.6.1 SendIPv6Message

This message is used to trigger the sending of an IPv6 message on a specified network interface.

Request (UtGn6Trigger TS ➔ UT):

0	1	2	3
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0x60	InterfaceLength	InterfaceName …	
SrcMacAddress			
…		DestMacAddress	
…			
PacketLength		IPv6Packet …	

Name	Length	Value
MessageType	1 byte	0x60
InterfaceLength	1 byte	Length of "InterfaceName" field
InterfaceName	InterfaceLength × 1 byte	Name of the interface on which to send the IPv6 packet
SrcMacAddress	6 bytes	Source MAC address
DestMacAddress	6 bytes	Destination MAC address
PacketLength	2 bytes	Length of the "IPv6Packet" field
IPv6Packet	Variable	IPv6 packet to be sent

Response (UtGn6TriggerResult UT ➔ TS):

0 0 1 2 3 4 5 6 7	1 0 1 2 3 4 5 6 7
MessageType = 0x61	Result

Name	Length	Value
MessageType	1 byte	0x81
Result	1 byte	0x00: Failure 0x01: Success

C.6.2 GetInterfaceInfos

This message is used by the Test System to retrieve the configuration of network interfaces on IUT.

Request (UtGn6GetInterfaceInfo TS ➔ UT):

0 0 1 2 3 4 5 6 7
MessageType = 0x62

Name	Length	Value
MessageType	1 byte	0x62

Response (UtGn6GetInterfaceInfoResult UT ➔ TS):

0 0 1 2 3 4 5 6 7	1 0 1 2 3 4 5 6 7	2 0 1 2 3 4 5 6 7	3 0 1 2 3 4 5 6 7
MessageType = 0x63	InterfaceCount	InterfaceLength[0]	InterfaceName[1]
…	AddressCount[0]	Addresses[0][0]	
…			
…			
…			
…		Addresses[0][1]	
…		InterfaceLength[1]	InterfaceName[1]
…	…		

Name		Length	Value
MessageType		1 byte	0x85
InterfaceCount		1 byte	Number of interface descriptors
Interface Descriptor	InterfaceLength	1 byte	Length of "InterfaceName" field
	InterfaceName	InterfaceLength × 1 byte	Name of the interface
	AddressCount	1 byte	Number of configured IPv6 address on the interface
	Addresses	AddressCount × 16 bytes	IPv6 addresses configured on interface

C.6.3 Gn6EventIndication

This message is used to check whether payload contained in GeoNetworking PDU has been transmitted to upper layer (CAM/DENM/IPv6).

Indication (UtGnEventInd UT ➔ TS):

<table>
<tr><td colspan="8">0</td><td colspan="8">1</td><td colspan="8">2</td><td colspan="8">3</td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td colspan="8">MessageType = 0x64</td><td colspan="8">InterfaceLength</td><td colspan="16">InterfaceName …</td></tr>
<tr><td colspan="16">PacketLength</td><td colspan="16">IPv6Packet</td></tr>
</table>

Name	Length	Value
MessageType	1 byte	0x64
InterfaceLength	1 byte	Length of "InterfaceName" field
InterfaceName	InterfaceLength × 1 byte	Name of the interface on the IPv6 packet has been received
PacketLength	2 bytes	Length of ' IPv6Packet' field
IPv6Packet	Variable	Received IPv6 packet

NOTE: Gn6 primitives are not yet supported by the ITS test suite.

C.7 BTP Upper Tester Primitives

C.7.1 GenerateBtpA

This message is used to trigger a BTP-A message.

Request (UtBtpTrigger_A TS ➔ UT):

<table>
<tr><td colspan="8">0</td><td colspan="8">1</td><td colspan="8">2</td><td colspan="8">3</td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td colspan="8">MessageType = 0x70</td><td colspan="16">DestPort</td><td colspan="8">SrcPort</td></tr>
<tr><td colspan="8">…</td></tr>
</table>

Name	Length	Value
MessageType	1 byte	0x70
DestPort	2 bytes	Destination port
SrcPort	2 bytes	Source port

Response (UtBtpTriggerResult UT ➔ TS):

<table>
<tr><td colspan="8">0</td><td colspan="8">1</td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td colspan="8">MessageType = 0x71</td><td colspan="8">Result</td></tr>
</table>

Name	Length	Value
MessageType	1 byte	0x71
Result	1 byte	0x00: Failure 0x01: Success

C.7.2 GenerateBtpB

This message is used to trigger a BTP-B message.

Request (UtBtpTrigger_B TS ➔ UT):

0	1	2	3
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0x72	DestPort		DestPortInfo
...			

Name	Length	Value
MessageType	1 byte	0x72
DestPort	2 bytes	Destination port
DestPortInfo	2 bytes	Destination port info

Response (UtBtpTriggerResult UT ➔ TS):

0	1
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0x71	Result

Name	Length	Value
MessageType	1 byte	0x71
Result	1 byte	0x00: Failure 0x01: Success

C.7.3 BtpEventIndication

This message is used to check whether payload contained in BTP PDU has been transmitted to upper layer.

Indication (UtBtpEventInd UT ➔ TS):

0	1	2	3
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0x73	PacketLength		Packet
...			

Name	Length	Value
MessageType	1 byte	0x73
PacketLength	2 bytes	Length of 'Packet' field
Packet	Variable	Packet's final payload

C.8 MAPEM/SPATEM Upper Tester Primitives

C.8.0 UtInitialize

The command UtInitialize is the same for both RLT (MAP) and TLM (SPaT) modules. When the IUT receives an UtInitialize message, it will start sending MAPEM.

C.8.1 MapemSpatemTrigger

When the IUT receives an UtMapemSpatemTrigger message, the IUT will start sending SPATEM.

This message is used to trigger a specific event.

Request (UtMapemSpatemTrigger TS ➔ UT):

0	1								2	3
0 1 2 3 4 5 6 7	0	1	2	3	4	5	6	7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0x7a	I	R	X	X	X	X	X	X	Event	IntersectionID
…	RegulatorySpeedLimit…									
…										

Name	Length	Value
MessageType	1 byte	0x7a
I	1 bit	0: IntersectionID to be ignored 1: IntersectionID to be used
R	1 bit	0: RegulatorySpeedLimit to be ignored 1: RegulatorySpeedLimit to be used
Event	1 byte	0 MAP Profile #1 (see notes below) 1 MAP Profile #2 (see notes below) 2 MAP Profile #3 (see notes below) 3 SPaT Profile #1 4 SPaT Profile #2 5 SPaT Profile #3 6 SPaT Profile #4 10 Pedestrian detected 20 Stop TLM service
IntersectionID	2 bytes	Intersection identifier
RegulatorySpeedLimit	1 bytes + 2 bytes	Set RegulatorySpeedLimit field (SpeedLimitType and Velocity)

Response (UtMapemSpatemTriggerResult UT ➔ TS):

0	1
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0x7b	Result

Name	Length	Value
MessageType	1 byte	0x7b
Result	1 byte	0x00: Failure 0x01: Success

NOTE 1: To keep the message simple, different MAPE message profiles are proposed here. Each profile is identified by a unique ID (see MessageType described above):

- Map Profile #1: This is a basic MAPE message with only one ingress road segment and one egress road segment, no intersections and one or more crosswalk. For each road segment, two connection points are set.

- Map Profile #2: It derives the Profile#1 by adding one intersection and traffic lights (to be used with SPaT profile 0).

- Map Profile #3: It derives the Profile#1 with huge data in order to test message fragmentation.

- Spat Profile #1: This is a basic SPaT message to be used with Map Profile #2.

- Spat Profile #2: This is a basic SPaT message (Spat Profile #1) containing optimal speed information.

- Spat Profile #3: This is a basic SPaT message (Spat Profile #1) containing greenwave and signal timing progression speed information.

- Spat Profile #4: This is a basic SPaT message (Spat Profile #1) containing queue and the length of available vehicular storage.

NOTE 2: Profile #1 is the default profile to be used to generate MAPE message when the RSU is in idle state.

C.8.2 MapemEventInd

This message is used to indicate reception of MAP information by IUT.

Indication (UtMapemEventInd UT ➔ TS):

```
0                       1                               2                               3
0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7
|MessageType = 0x7c     |PacketLength                                   |Packet                 :
:                                    ...                                                        |
```

Name	Length	Value
MessageType	1 byte	0x7c
PacketLength	2 bytes	Length of 'MapemPdu' field
Packet	Variable	Received MAPEM

C.8.3 SpatemEventInd

This message is used to indicate reception of SPaT information by IUT.

Indication (UtSpatemEventInd UT ➔ TS):

```
0                       1                               2                               3
0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7
|MessageType = 0x7d     |PacketLength                                   |Packet                 :
:                                    ...                                                        |
```

Name	Length	Value
MessageType	1 byte	0x7d
PacketLength	2 bytes	Length of 'SpatemPdu' field
Packet	Variable	Received SPATEM

C.9 IVIM Upper Tester Primitives

C.9.1 GenerateIvimEvent

This message is used to trigger a specific event.

Request (UtIvimTrigger TS ➔ UT):

```
0                       1                               2                               3
0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7  0  1  2  3  4  5  6  7
|MessageType = 0x80     |F  |V  |T  |I  |L  |Z  |D  |A  |R  |G  |Y  |X  |X  |X  |X  |X  |                       |
|ValidFrom                                                                                          |   |
|ValidTo                                                                                            |   |
|…                                              |RepetitionInterval                                 |   |
|…                                                                                                  |   |
```

Name	Length	Value
MessageType	1 byte	0x80
F	1 bit	0: ValidFrom to be ignored 1: ValidFrom to be used
V	1 bit	0: ValidTo to be ignored 1: ValidTo to be used
T	1 bit	0: RepetitionInterval to be ignored 1: RepetitionInterval to be used
I	1 bit	0: ZoneId to be ignored 1: ZoneId to be used
L	1 bit	0: Lane to be ignored 1: Lane to be used
Z	1 bit	0: RelevanceZoneId to be ignored 1: RelevanceZoneId to be used
D	1 bit	0: DetectionZoneId to be ignored 1: DetectionZoneId to be used
A	1 bit	0: DriverAwarenesZoneIds to be ignored 1: DriverAwarenesZoneIds to be used
R	1 bit	0: ItsRrid to be ignored 1: ItsRrid to be used
G	1 bit	0: Direction to be ignored 1: Direction to be used
Y	1 bit	0: layoutId to be ignored 1: layoutId to be used
X	1 bit	Reserved
ValidFrom	6 bytes	The ValidFrom date/time in milliseconds
ValidTo	6 bytes	The ValidTo date/time in milliseconds
RepetitionInterval	6 bytes	The repetition interval value on repetition activation in millisecond, 0 otherwise
ZoneIds	1 byte+ N bytes	The ZoneId list contains the size of the list and zoneId values. (020102 - 2 zoneIds present - zoneId 1 and zoneId 2)
Lane	1byte	The LanePosition present in integer(values from -1 to 14)
RelavanceZoneId	1 byte+ N bytes	The RelavanceZoneId list contains the size of the list and zoneId values. (020102 - 2 zoneIds present - zoneId 1 and zoneId 2)
DetectionZoneId	1 byte+ N bytes	The DetectionZoneId list contains the size of the list and zoneId values. (020102 - 2 zoneIds present - zoneId 1 and zoneId 2)
DriverAwarenesZoneId	1 byte+ N bytes	The DriverAwarenesZoneId list contains the size of the list and zoneId values. (020102 - 2 zoneIds present - zoneId 1 and zoneId 2)
ItsRrid	1 byte	The ItsRrid contains VarLengthNumber (1 octet length) or with extensions more octets
Direction	1 byte	The direction is integer. (values from 0-3)
layoutId	1 byte	The layoutId is integer.

NOTE 1: Several zoneID will be used.

All these zones do not overlap.

Here is an example for two zones:

```
glc
   referencePosition
      latitude: Unknown (435512756)
      longitude: Unknown (103002535)
      positionConfidenceEllipse
         semiMajorConfidence: oneCentimeter (1)
         semiMinorConfidence: oneCentimeter (1)
         semiMajorOrientation: wgs84North (0)
      altitude
         altitudeValue: referenceEllipsoidSurface (0)
         altitudeConfidence: unavailable (15)
   parts: 2 items
      Item 0
   GlcPart
      zoneId: 1
      zone: segment (0)
         segment
            line: deltaPositions (0)
               deltaPositions: 5 items
                  Item 0
                     DeltaPosition
                        deltaLatitude: Unknown (-227)
                        deltaLongitude: Unknown (-5)
                  Item 1
                     DeltaPosition
                        deltaLatitude: Unknown (-146)
                        deltaLongitude: Unknown (-1187)
                  Item 2
                     DeltaPosition
                        deltaLatitude: Unknown (-109)
                        deltaLongitude: Unknown (-1664)
                  Item 3
                     DeltaPosition
                        deltaLatitude: Unknown (-141)
                        deltaLongitude: Unknown (-1295)
                  Item 4
                     DeltaPosition
                        deltaLatitude: Unknown (-233)
                        deltaLongitude: Unknown (-2153)
Item 1
   GlcPart
      zoneId: 2
      zone: segment (0)
         segment
            line: deltaPositions (0)
               deltaPositions: 5 items
                  Item 0
                     DeltaPosition
                        deltaLatitude: Unknown (-179)
                        deltaLongitude: Unknown (356)
                  Item 1
                     DeltaPosition
                        deltaLatitude: Unknown (100)
                        deltaLongitude: Unknown (891)
                  Item 2
                     DeltaPosition
                        deltaLatitude: Unknown (94)
                        deltaLongitude: Unknown (953)
                  Item 3
                     DeltaPosition
                        deltaLatitude: Unknown (87)
                        deltaLongitude: Unknown (930)
                  Item 4
                     DeltaPosition
                        deltaLatitude: Unknown (105)
                        deltaLongitude: Unknown (930)
```

NOTE 2: Road Works Warning container will be included when DriverAwarenesZoneId is specified.

Response (UtIvimTriggerResult UT ➔ TS):

<table>
<tr><td colspan="8">0</td><td colspan="8">1</td><td colspan="8">2</td><td colspan="8">3</td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td colspan="8">MessageType = 0x81</td><td colspan="8">Result</td><td colspan="16">IviIdentificationNumber</td></tr>
</table>

Name	Length	Value
MessageType	1 byte	0x81
Result	1 byte	0x00: Failure 0x01: Success
IviIdentificationNumber	2 bytes	Value of the IviIdentificationNumber of the generated IVI message when Result field is true, ignored otherwise

C.9.2 UpdateIvimEvent

This message is used to trigger a specific event.

Request (UtIvimUpdate TS ➔ UT):

<table>
<tr><td colspan="8">0</td><td colspan="8">1</td><td colspan="8"></td><td colspan="8"></td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td colspan="8">MessageType = 0x82</td><td>F</td><td>V</td><td>T</td><td>C</td><td>X</td><td>X</td><td>X</td><td>X</td><td colspan="16">IviIdentificationNumber</td></tr>
<tr><td colspan="32">ValidFrom</td></tr>
<tr><td colspan="16">…</td><td colspan="16">ValidTo</td></tr>
<tr><td colspan="32">…</td></tr>
<tr><td colspan="32">RepetitionInterval</td></tr>
<tr><td colspan="16">…</td><td colspan="16">connectedIviStructures</td></tr>
<tr><td colspan="32">…</td></tr>
</table>

Name	Length	Value
MessageType	1 byte	0x82
F	1 bit	0: ValidFrom to be ignored 1: ValidFrom to be used
V	1 bit	0: ValidTo to be ignored 1: ValidTo to be used
T	1 bit	0: RepetitionInterval to be ignored 1: RepetitionInterval to be used
C		0: ConnectedIviStructures to be ignored 1: ConnectedIviStructures to be used
IviIdentificationNumber	2 bytes	Value of the IviIdentificationNumber of the IVI message to be updated
ValidFrom	6 bytes	The ValidTo date/time in milliseconds
ValidTo	6 bytes	The ValidTo date/time in milliseconds
RepetitionInterval	6 bytes	The repetition interval value on repetition activation or a new repetition interval value in second, 0 to deactivate it
ConnectedIviStructures	1 byte + 2 x N bytes	List of other iviIdentificationNumber identifying other IVI Structures of the same authority which are connected to the IVI, two byte per iviIdentificationNumber

Response (UtIvimUpdateResult UT ➔ TS):

<table>
<tr><td colspan="8">0</td><td colspan="8">1</td><td colspan="8">2</td><td colspan="8">3</td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td colspan="8">MessageType = 0x83</td><td colspan="8">Result</td><td colspan="16">IviIdentificationNumber</td></tr>
</table>

Name	Length	Value
MessageType	1 byte	0x83
Result	1 byte	0x00: Failure 0x01: Success
IviIdentificationNumber	2 bytes	Value of the IviIdentificationNumber of the updated IVI message when Result field is true, ignored otherwise

C.9.3 TerminateIvimEvent

This message is used to trigger a specific event.

Request (UtIvimTerminate TS ➔ UT):

0	1	2
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0x84	IviIdentificationNumber	

Name	Length	Value
MessageType	1 byte	0x84
IviIdentificationNumber	2 bytes	Value of the IviIdentificationNumber of the IVI message to be terminated

Response (UtIvimTerminationResult UT ➔ TS):

0	1	2	
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0x85	Result	IviIdentificationNumber	

Name	Length	Value
MessageType	1 byte	0x85
Result	1 byte	0x00: Failure 0x01: Success
IviIdentificationNumber	2 bytes	Value of the IviIdentificationNumber of the terminated IVI message when Result field is true, ignored otherwise

C.9.4 IvimEventInd

This message is used to indicate reception of IVI information by IUT.

Indication (UtIvimEventInd UT ➔ TS):

0	1	2	3
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0x86	PacketLength		Packet
...			

Name	Length	Value
MessageType	1 byte	0x86
PacketLength	2 bytes	Length of 'IvimPdu' field
Packet	Variable	Received IVIM

C.10 SREM/SSEM Upper Tester Primitives

C.10.1 GenerateSremEvent

This message is used to trigger a specific event.

Request (UtSremTrigger TS ➔ UT):

0	1	2	
` 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	
MessageType = 0xA0	BasicVehicleRole	RequestImportanceLevel	IntersectionID
…			

Name	Length	Value
MessageType	1 byte	0xA0
BasicVehicleRole	1 bytes	Vehicle role
RequestImportanceLevel	1 bytes	Request importance level value
IntersectionID	2 bytes	Intersection identifier

Response (UtSremTriggerResult UT ➔ TS):

0	1	2
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0xA1	Result	RequestID

Name	Length	Value
MessageType	1 byte	0xA1
Result	1 byte	0x00: Failure 0x01: Success

C.10.2 UpdateSremEvent

This message is used to trigger a specific event.

Request (UtSremUpdate TS ➔ UT):

0	1	2	
` 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	
MessageType = 0xA2	BasicVehicleRole	RequestImportanceLevel	IntersectionID
…			

Name	Length	Value
MessageType	1 byte	0xA2
BasicVehicleRole	1 bytes	Vehicle role
RequestImportanceLevel	1 bytes	Request importance level value
IntersectionID	2 bytes	Intersection identifier

Response (UtSremUpdateResult UT ➔ TS):

0	1	
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	
MessageType = 0xA3	Result	

Name	Length	Value
MessageType	1 byte	0xA3
Result	1 byte	0x00: Failure 0x01: Success

C.10.3 TerminateSremEvent

This message is used to terminate an existing SREM event.

Request (UtSrememTermination TS ➔ UT):

0	1	2	3
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0xA4	MsgCount		

Name	Length	Value
MessageType	1 byte	0xC4
MsgCount	1 bytes	Message count

Response (UtSremTerminationResult UT ➔ TS):

0 0 1 2 3 4 5 6 7	1 0 1 2 3 4 5 6 7	2 0 1 2 3 4 5 6 7	3 0 1 2 3 4 5 6 7
MessageType = 0xA5	Result		

Name	Length	Value
MessageType	1 byte	0xC5
Result	1 byte	0x00: Failure 0x01: Success

C.10.4 SremEventInd

This message is used to indicate reception of SREM information by IUT.

Indication (UtSremEventInd UT ➔ TS):

0 0 1 2 3 4 5 6 7	1 0 1 2 3 4 5 6 7	2 0 1 2 3 4 5 6 7	3 0 1 2 3 4 5 6 7
MessageType = 0xA6	PacketLength		Packet
…			

Name	Length	Value
MessageType	1 byte	0xAF
PacketLength	2 bytes	Length of 'SremPdu' field
Packet	Variable	Received SREM

C.10.5 SsemEventInd

This message is used to indicate reception of IVI information by IUT.

Indication (UtSsemEventInd UT ➔ TS):

0 0 1 2 3 4 5 6 7	1 0 1 2 3 4 5 6 7	2 0 1 2 3 4 5 6 7	3 0 1 2 3 4 5 6 7
MessageType = 0xA7	PacketLength		Packet
	...		

Name	Length	Value
MessageType	1 byte	0xB0
PacketLength	2 bytes	Length of 'SsemPdu' field
Packet	Variable	Received SSEM

C.11 RTCMEM Upper Tester Primitives

C.11.1 GenerateRtcmemEvent

This message is used to trigger a specific event.

Request (UtRtcmemTrigger TS ➔ UT):

0	1	2	3
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0xC0	T P H X X X X X	Message count	RTCM version
List of RTCM messages to be transmitted			
…			
Timestamp			
		Anchor Point	
…			
RTCMEM header			

0	1	2
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0xC0	Flags	MsgCount

Name	Length	Value
MessageType	1 byte	0xC0
T	6 bytes	Timestamp
P		Anchor Point
H		RTCM Header
MsgCount	1 bytes	Message count
RTCM_Revision	1 bytes	RTCM version
RTCMmessageList	1 byte + (1byte + n bytes)	List of RTCM messages to be transmitted
Timestamp	4 bytes	Minutes of current UTC year
Anchor Point	8 bytes	Longitude (4 bytes) and Latitude (4 bytes)
RTCMEM header	7 bytes	RTCM header details (GNSS status: 1 byte + Antenna offset: 3 x 2 bytes)

Response (UtRtcmemTriggerResult UT ➔ TS):

0	1	2
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
MessageType = 0xC1	Result	

Name	Length	Value
MessageType	1 byte	0xC1
Result	1 byte	0x00: Failure 0x01: Success

C.11.2 UpdateRtcmemEvent

Void.

C.11.3 TerminateRtcmemEvent

This message is used to terminate an existing RTCMEM event.

Request (UtRtcmemTermination TS ➔ UT):

```
0                       1                       2                       3
0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
|MessageType = 0xC4     |MsgCount               |
```

Name	Length	Value
MessageType	1 byte	0xC4
MsgCount	1 bytes	Message count

Response (UtRtcmemTerminationResult UT ➔ TS):

```
0                       1                       2                       3
0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
|MessageType = 0xC5     |Result                 |
```

Name	Length	Value
MessageType	1 byte	0xC5
Result	1 byte	0x00: Failure 0x01: Success

C.11.4 RtcmemEventIndication

This message is used to indicate reception of RTCMEM information by IUT.

Indication (UtRtcmemEventInd UT ➔ TS):

```
0                       1                       2                       3
0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
|MessageType = 0xC6     |RtcmemPduLength                                |RtcmemPdu
                                       ...                                              |
```

Name	Length	Value
MessageType	1 byte	0xC6
RtcmemPduLength	2 bytes	Length of 'RtcmemPdu' field
RtcmemPdu	Variable	Received RTCMEM

C.12 PKI Upper Tester Primitives

C.12.1 GenerateInnerEcRequest

This message is used to trigger an InnerEcRequest event.

Request (UtPkiTrigger TS ➔ UT):

```
0                       1                       2
0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
|MessageType = 0xD0     |
```

Name	Length	Value
MessageType	1 byte	0xD0

Response (UtPkiTriggerResult UT ➔ TS):

0								1								2							
0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
MessageType = 0xD2								Result															

C.12.2 GenerateInnerAtRequest

This message is used to trigger an InnerAtRequest event.

Request (UtPkiTrigger TS ➔ UT):

0								1								2							
0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
MessageType = 0xD1																							

Name	Length	Value
MessageType	1 byte	0xD1

Response (UtPkiTriggerResult UT ➔ TS):

0								1								2							
0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
MessageType = 0xD2								Result															

Annex D: Example of Test Platform implementation

The test platform for validating ITS conformance test suites has been developed using the following tools and components:

- Standard PC equipped with two Ethernet network cards (It is possible to use the same single network card for both tasks, but it is less practical and gives less flexibility).
  One network card (Mac address: 00-A0-24-AD-56-FF) is used to communicate with radio device.
  The second one (Mac address: 00-50-56-C0-00-08) is used to establish upper tester link with SUT and is configured with IP address 192.168.56.1/24.

- Windows™ 7 Professional operating system (64 bits)
  No special requirement concerning operating system. Theoretically, the platform can be used on Linux® based operating systems, as it is OS independent.

- Spirent TTworkbench Basic v22 with ASN.1 plugins
  ASN.1 plugins are necessary for CAM and DENM codecs. Any other TTCN-3 test tool would be suitable with minimum adaptation as the test platform is tool independent.

- Elvior TestCast v22.

- Java™ JDK 1.8.x
  All the software used in the test platform have been developed using Java™ language.

- JnetPcap 1.4.x
  This library is used for capturing and injecting raw Ethernet packets. It is a direct dependency of EthernetLayer module. For easy setup the `jnetpcap.dll` file needs to be installed in `C:\Windows\System\` folder and the jnetpcap.jar needs to be installed in `C:\Windows\Sun\Java\lib\ext\` folder or equivalent. By doing this, no specific setting will be required to include JnetPcap library when building Test Adapter.

- Access layer adapter:

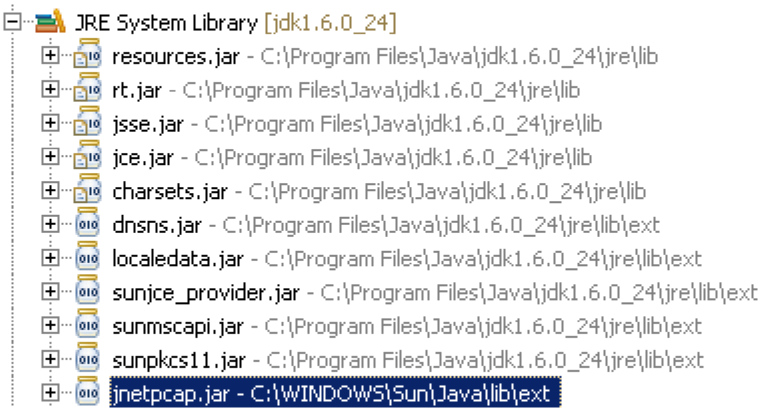
  - G5 switch
    This device provides G5 connectivity to the test platform. This device features a G5 radio interface used to communicate with SUT and Ethernet interface that is connect to the test platform PC in order to transfer G5 packets to be sent and received via the radio interface.

  - LTE C-V2X switch
    This device provides the LTE C-V2X connectivity, allowing to Test System to communicate with LTE C-V2X devices.

Before running successfully any test, a certain number of settings need to be verified in TTworkbench:

- Project needs to be set for using Java™ JDK 1.8.x. Note that JnetPcap should automatically appear in the library list/

NOTE 1: Figure D.1 contains still the Java™ JDK 1.6.0.

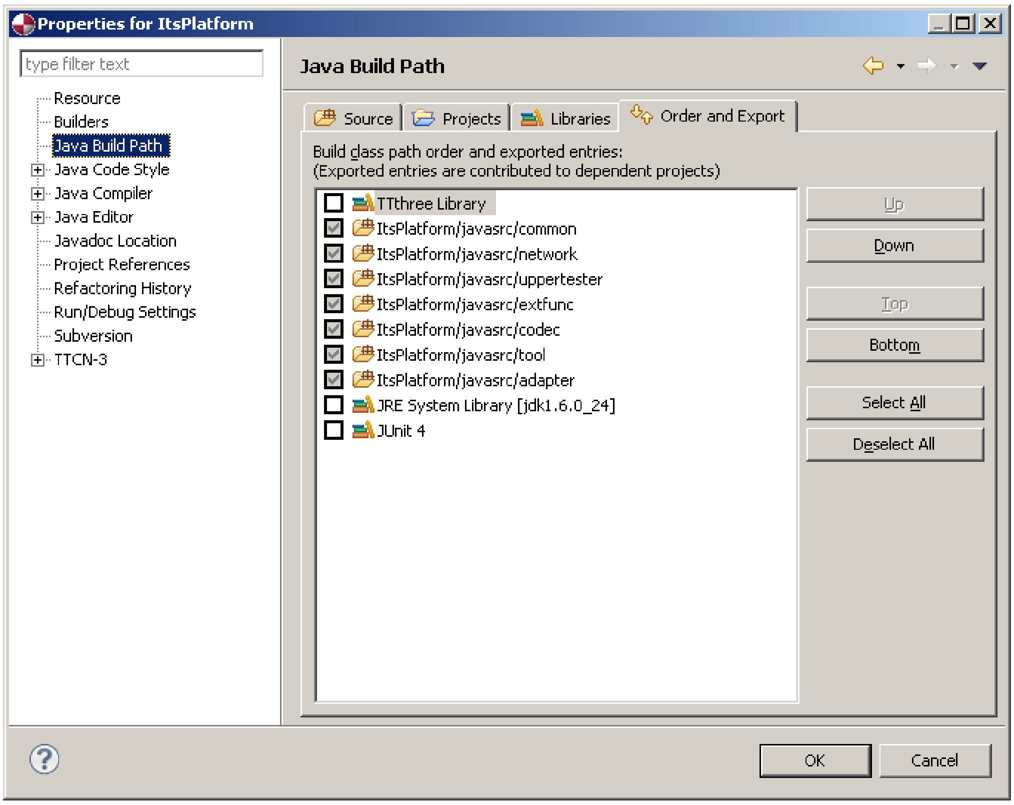


```
JRE System Library [jdk1.6.0_24]
  resources.jar - C:\Program Files\Java\jdk1.6.0_24\jre\lib
  rt.jar - C:\Program Files\Java\jdk1.6.0_24\jre\lib
  jsse.jar - C:\Program Files\Java\jdk1.6.0_24\jre\lib
  jce.jar - C:\Program Files\Java\jdk1.6.0_24\jre\lib
  charsets.jar - C:\Program Files\Java\jdk1.6.0_24\jre\lib
  dnsns.jar - C:\Program Files\Java\jdk1.6.0_24\jre\lib\ext
  localedata.jar - C:\Program Files\Java\jdk1.6.0_24\jre\lib\ext
  sunjce_provider.jar - C:\Program Files\Java\jdk1.6.0_24\jre\lib\ext
  sunmscapi.jar - C:\Program Files\Java\jdk1.6.0_24\jre\lib\ext
  sunpkcs11.jar - C:\Program Files\Java\jdk1.6.0_24\jre\lib\ext
  jnetpcap.jar - C:\WINDOWS\Sun\Java\lib\ext
```

Figure D.1

- Test Adapter and Codecs source folders need to be declared in project's Java™ Build Path.

NOTE 2: Figure D.2 contains still the Java™ JDK 1.6.0.



Build class path order and exported entries:
(Exported entries are contributed to dependent projects)

☐ TTthree Library
☑ ItsPlatform/javasrc/common
☑ ItsPlatform/javasrc/network
☑ ItsPlatform/javasrc/uppertester
☑ ItsPlatform/javasrc/extfunc
☑ ItsPlatform/javasrc/codec
☑ ItsPlatform/javasrc/tool
☑ ItsPlatform/javasrc/adapter
☐ JRE System Library [jdk1.6.0_24]
☐ JUnit 4

Figure D.2

- Test Adapter and Codecs will then be automatically compiled if "Build automatically" option is set.

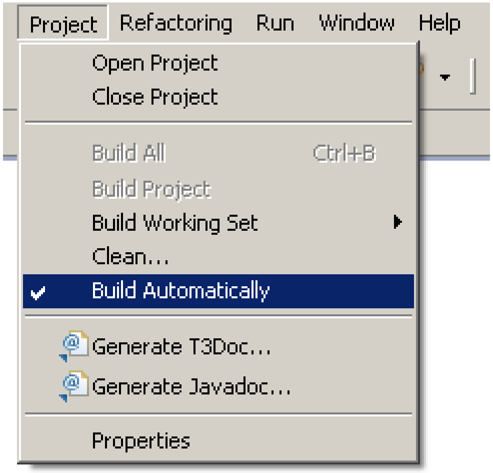


Figure D.3

- Alternatively, Test Adapter and Codecs precompiled libraries need to be referenced as external libraries.
- TTCN-3 test suites and ASN.1 definitions are copied to the project and declared as TTCN-3 source folders.

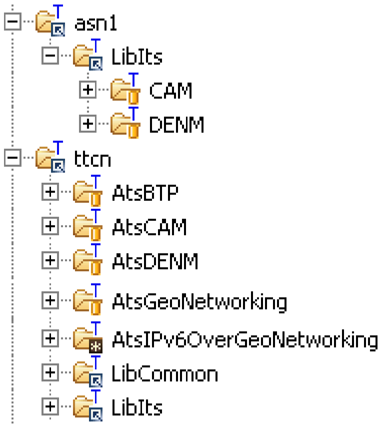


Figure D.4

- The test suites are compiled using the "Rebuild All" button.

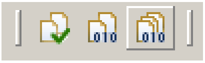
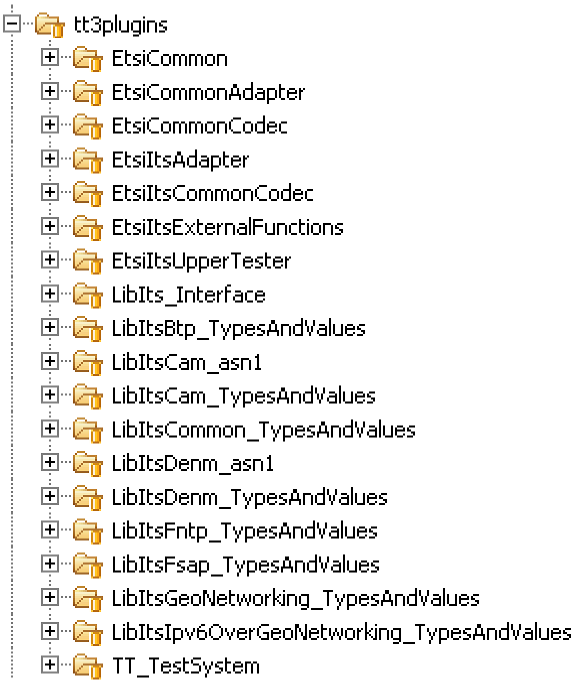


Figure D.5

- TTCN-3 plugins are configured using the provided xml files.



```
tt3plugins
    EtsiCommon
    EtsiCommonAdapter
    EtsiCommonCodec
    EtsiItsAdapter
    EtsiItsCommonCodec
    EtsiItsExternalFunctions
    EtsiItsUpperTester
    LibIts_Interface
    LibItsBtp_TypesAndValues
    LibItsCam_asn1
    LibItsCam_TypesAndValues
    LibItsCommon_TypesAndValues
    LibItsDenm_asn1
    LibItsDenm_TypesAndValues
    LibItsFntp_TypesAndValues
    LibItsFsap_TypesAndValues
    LibItsGeoNetworking_TypesAndValues
    LibItsIpv6OverGeoNetworking_TypesAndValues
    TT_TestSystem
```

Figure D.6

- Test adapter parameters are adjusted in `taconfig.xml` file:

```
<parameter id="taParams">
      <parameter id="DEBUG_ENABLED" value="ALL"/>
      <parameter id="camPort" value="BTP/GN/ETH"/>
      <parameter id="denmPort" value="BTP/GN/ETH"/>
      <parameter id="btpPort" value="GN/ETH"/>
      <parameter id="geoNetworkingPort" value="ETH"/>
      <parameter id="ipv6OverGeoNetworkingPort" value="Debug"/>
      <parameter id="CamUpperTester" value="Operator"/>
      <parameter id="CamUpperTesterSettings" value=""/>
      <parameter id="DenmUpperTester" value="Operator"/>
      <parameter id="DenmUpperTesterSettings" value=""/>
      <parameter id="BtpUpperTester" value="Operator"/>
      <parameter id="BtpUpperTesterSettings" value=""/>
      <parameter id="GnUpperTester" value="Generic"/>
      <parameter id="GnUpperTesterSettings"
            value="NwtaTrigger:192.168.56.10:1600:1601"/>
      <parameter id="LocalEthernetMAC" value="00A024AD56FF"/>
      <parameter id="IutEthernetTypeValue" value="0x0707"/>
      <parameter id="LinkLayer_MTC" value="BABEBABE0000"/>
      <parameter id="LinkLayer_NodeA" value="BABEBABE0001"/>
      <parameter id="LinkLayer_NodeB" value="BABEBABE0002"/>
      <parameter id="LinkLayer_NodeC" value="BABEBABE0003"/>
      <parameter id="LinkLayer_NodeD" value="BABEBABE0004"/>
      <parameter id="Gn6RemoteAdapterIp" value="192.168.56.10"/>
      <parameter id="Gn6RemoteAdapterPort" value="42000"/>
</parameter>
```

Table D.1 summarizes the authorized values for these parameters.

Table D.1: Test Adapter Parameters

Parameter	Description	Allowed values
DEBUG_ENABLED	Indicates whether Codecs and Test Adapter produce debugging logs	ALL, NONE, OFF
camPort	Defines the lower stack of CamPort	Any combination of valid layer identifier separated by "/" symbol: • ETH • BTP • GN • UdpIp • Debug (pseudo layer that dumps packet to console) • Loopback (pseudo layer that reinjects the packets)
denmPort	Defines the lower stack of CamPort	
btpPort	Defines the lower stack of CamPort	
geoNetworkingPort	Defines the lower stack of CamPort	
ipv6OverGeoNetworkingPort	Defines the lower stack of CamPort	
CamUpperTester DenmUpperTester BtpUpperTester GnUpperTester	Selects the type of Upper tester to be used for each test suite	Operator, Yes, Generic
CamUpperTesterSettings DenmUpperTesterSettings BtpUpperTesterSettings GnUpperTesterSettings	Defines Upper Tester specific settings like remote IP addresses, UDP ports, etc.	Upper tester specific
LocalEthernetMAC	MAC Address of the Ethernet card used to communicate with radio equipment	Hexstring representation of Mac Address without separator
IutEthernetTypeValue	Ethertype value to be used for sending and capturing packets	Integer 0 to 65 635. Should be 0x0707
LinkLayer_MTC LinkLayer_NodeA LinkLayer_NodeB LinkLayer_NodeC LinkLayer_NodeD	MAC addresses used by simulated ITS nodes	Hexstring representation of Mac Address without separator
Gn6RemoteAdapterIp	IP Address of GN6 Remote Adapter	Standard IP address notation
Gn6RemoteAdapterPort	UDP port of GN6 Remote Adapter	Integer 0 to 65 635

Annex E: Complete Test Adapter class diagram

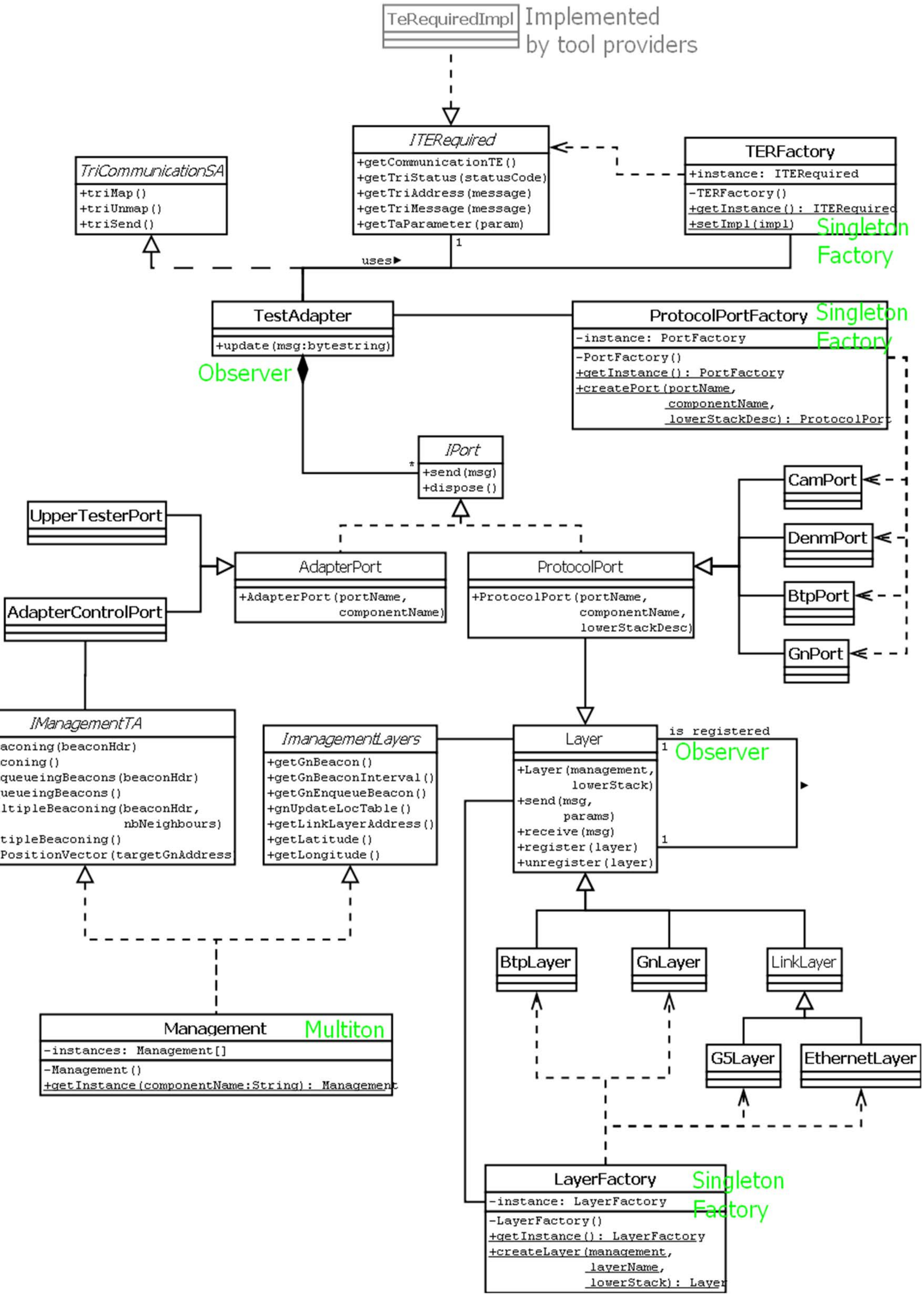


Figure E.1: Test adapter complete class diagram

Annex F: Bibliography

This annex lists all test specifications which were integrated with the Conformance Validation Framework:

- ETSI TS 102 868-1 (V1.2.1): "Intelligent Transport Systems (ITS); Testing; Conformance test specification for Co-operative Awareness Messages (CAM); Part 1: Test requirements and Protocol Implementation Conformance Statement (PICS) proforma".
- ETSI TS 102 868-2 (V1.2.1): "Intelligent Transport Systems (ITS); Testing; Conformance test specification for Co-operative Awareness Messages (CAM); Part 2: Test Suite Structure and Test Purposes (TSS&TP)".
- ETSI TS 102 869-1 (V.1.2.1): "Intelligent Transport Systems (ITS); Testing; Conformance test specification for Of Decentralized Environmental Notification basic Service (DENM); Part 1: Test requirements and Protocol Implementation Conformance Statement (PICS) proforma".
- ETSI TS 102 869-2 (V.1.3.1): "Intelligent Transport Systems (ITS); Testing; Conformance test specification for Of Decentralized Environmental Notification basic Service (DENM); Part 2: Test Suite Structure and Test Purposes (TSS&TP)".
- ETSI TS 102 870-1 (V.1.3.1): "Intelligent Transport Systems (ITS); Testing; Conformance test specifications for Geonetworking Basic Transport Protocol (BTP); Part 1: Test requirements and Protocol Implementation Conformance Statement (PICS) proforma".
- ETSI TS 102 870-2 (V.1.1.1): "Intelligent Transport Systems (ITS); Testing; Conformance test specifications for Geonetworking Basic Transport Protocol (BTP); Part 2: Test Suite Structure and Test Purposes (TSS&TP)".
- ETSI TS 102 859-1 (V.1.2.1): "Intelligent Transport Systems (ITS); Testing; Conformance test specifications for Transmission of IP packets over GeoNetworking; Part 1: Test requirements and Protocol Implementation Conformance Statement (PICS) proforma".
- ETSI TS 102 859-2 (V.1.2.1): "Intelligent Transport Systems (ITS); Testing; Conformance test specifications for Transmission of IP packets over GeoNetworking; Part 2: Test Suite Structure and Test Purposes (TSS&TP)".
- ETSI EG 202 798 (V1.1.1): "Intelligent Transport Systems (ITS); Testing; Framework for conformance and interoperability testing".
- JNetPcap library.

NOTE: Available at https://sourceforge.net/projects/jnetpcap/.

- ETSI ES 201 873-1 (V4.5.1): "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 1: TTCN-3 Core Language".
- ETSI ES 201 873-6 (V4.5.1): "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 6: TTCN-3 Control Interface (TCI)".

History

Document history		
V1.1.1	November 2012	Publication
V1.2.1	May 2014	Publication
V1.3.1	July 2015	Publication
V1.4.1	March 2017	Publication
V1.5.1	March 2022	Publication