3GPP TSG-T1 Meeting #10
Copenhagen, Denmark, 8th – 9th February 2001

Tdoc T1-010094

Presentation of Specification to TSG or WG

Presentation to: TSG T WG Meeting #11

Document for presentation: TS 34.123-3, Version 1.0.0

Presented for: Information

Abstract of document:

This specification is the third part of a multi-part specification containing, in TTCN, a suite of conformance tests for 3rd Generation User Equipment (UE). The tests are intended to ensure that User Equipment for 3GPP systems conform to the relevant 3GPP Technical Specifications.

The other parts of the multi-part specification are TS 34.123-2, which contains a pro-forma for the Implementation Conformance Statement (ICS), and TS 34.123-1, which contains a prose description of the test cases.

Changes since last presentation to TSG T Meeting #10:

First presentation

Outstanding Issues:

This specification contains 220 test cases of an expected final total of about 650.

Contentious Issues:

These test cases need to be verified by the industry, and changes requested to TSG-T1 where needed.

3GPP TS 34.123-3 V1.0.0 (2001-02)

Technical Specification

3rd Generation Partnership Project;
Technical Specification Group Terminal;
User Equipment (UE) conformance specification;
Part 3: Abstract Test Suites
(Release 1999)

Keywords
ATS, Terminal, Radio, Mobile, PLMN

3GPP

Postal address

3GPP support office address
650 Route des Lucioles - Sophia Antipolis
Valbonne - FRANCE
Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Internet
http://www.3gpp.org

Contents

Foreword

This Technical Specification has been produced by the 3rd Generation Partnership Project (3GPP).

The contents of the present document are subject to continuing work within the TSG and may change following formal TSG approval. Should the TSG modify the contents of the present document, it will be re-released by the TSG with an identifying change of release date and an increase in version number as follows:

Version x.y.z

where:

- x the first digit:
  - 1 presented to TSG for information;
  - 2 presented to TSG for approval;
  - 3 or greater indicates TSG approved document under change control.
- y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.
- z the third digit is incremented when editorial only changes have been incorporated in the document.

Introduction

The present document is 3rd part of a multi-part conformance test specification for UE. The specification contains a TTCN design frame work and the detailed test specifications in TTCN for UE at the Uu interface.

TS 34.123-1 (part 1) [1] contains a conformance test description in prose for UE at the Uu interface.

TS 34.123-2 (part 2) [2] contains a pro-forma for the UE Implementation Conformance Statement (ICS).

1 Scope

The present document specifies the protocol conformance testing in TTCN for the 3GPP User Equipment (UE) at the Uu interface.

The document is the 3rd part of a multi-part test specification, TS 34.123. The following information can be found in this part:

- The overall test suite structure;
- The testing architecture
- The test methods and PCO definitions;
- The test configurations;
- The design principles, assumptions, and used interfaces to the TTCN tester;
- TTCN styles and conventions,
- The partial PIXIT proforma;
- The TTCN.MP and TTCN.GR forms for the mentioned protocols tests.

The Abstract Test Suites designed in the document are based on the test cases specified in prose [1].

2 References

The following documents contain provisions, which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies.

[1] 3GPP TS 34.123-1: "UE Conformance Specification, Part 1 - Protocol conformance specification".

[2] 3GPP TS 34.123-2: "UE Conformance Specification, Part 2 - ICS proforma".

[3] 3GPP TS 34.108: "Common Test Environments for UE Conformance Testing".

[4] 3GPP TS 34.109: "Logical Test Interface (TDD and FDD)".

[5] 3GPP TR 21.905: "Vocabulary for 3GPP Specifications".

[6] 3GPP TS 23.003: "Numbering, Addressing and Identification".

[7] 3GPP TS 23.101: "General UMTS Architecture".

[8] 3GPP TS 24.007: "Mobile Radio Interface Signalling Layer 3; General Aspects".

[9] 3GPP TS 24.008: "Mobile Radio Interface Layer 3 specification; Core Network Protocols -Stage 3".

[10] 3GPP TS 24.011: "PP SMS support on the Mobile radio Interface".

[11] 3GPP TS 24.012: "SMSCB support on the Mobile radio Interface".

[12] 3GPP TS 25.214: "FDD; physical layer procedures".

[13] 3GPP TS 25.224: "TDD; physical layer procedures".

[14] 3GPP TS 25.301: "Radio Interface Protocol Architecture".

[15] 3GPP TS 25.303: "Interlayer Procedures in Connected Mode".

[16] 3GPP TS 25.304: "UE Procedure in Idle Mode and Procedures in Connected Mode".

[17] 3GPP TS 25.321: "MAC Protocol Specification".

[18] 3GPP TS 25.322: "Radio Link Control (RLC) Protocol Specification".

[19] 3GPP TS 25.323: "Packet Data Convergence Protocol (PDCP) Specification".

[20] 3GPP TS 25.324: "Radio Interface for Broadcast and Multicast Services".

[21] 3GPP TS 25.331: "Radio Resource Control (RRC) Protocol Specification".

[22] 3GPP TS 27.005: "Use of Data Terminal Equipment - Data Circuit terminating; Equipment (DTE - DCE) interface for Short Message Service (SMS) and Cell Broadcast Service (CBS)"

[23] 3GPP TS 27.007: "AT command set for User Equipment (UE)".

[24] 3GPP TS 27.060: "Mobile Stations (MS) supporting packet switched services".

[25] 3GPP TS 33.102: "Security architecture".

[26] 3GPP TS 51.010-1: "Mobile Station (MS) Conformance Specification".

[27] ETSI TR 101 666 (V1.0.0): "The Tree and Tabular Combined Notation (TTCN) (Ed. 2++)".

3 Definitions, symbols and abbreviations

3.1 Definitions

For the purposes of the present document, the following terms and definitions apply:

tbd.

3.2 Symbols

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

<symbol> <Explanation>

3.3 Abbreviations

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

<ACRONYM> <Explanation>

4 Requirements on the TTCN development

A number of requirements are identified for the development and production of TTCN specification for 3GPP UE at Uu interface.

1. Top-down design, following 3GPP TS 34.123-1[1], 3GPP TS 34.108[3] and 3GPP TS 34.109[4].

2. A unique testing architecture and test method for testing all protocol layers of UE,
3. Uniform TTCN style and naming conventions,
4. Improve TTCN readability,
5. Using TTCN-2++ (ETSI TR 101 666 [27]) for R99, avoid the use of the TTCN 2 features TTCN 3 does not support.
6. TTCN specification feasible, implementable and compilable,
7. Test cases shall be designed in a way for easily adaptable, upwards compatible with the evolution of the 3GPP core specifications and the future Releases.
8. The test declarations, data structures and data values shall be largely reusable.
9. Modularity and modular working method,
10. NAS ATS should be designed being independent from the radio access technologies.
11. Minimising the requirements of intelligence on the emulators of the lower testers. Especially the functionality of the RRC emulator in the TTCN tester should be reduced and simplified, the behaviours should be standardised as the TTCN RRC test steps in the TTCN modular library.
12. Giving enough design freedom to the test equipment manufacturers.
13. Maximising reuse of ASN.1 definitions from the relevant core specifications.

In order to fulfil these requirements and to ensure the investment of the test equipment manufacturers having a stable testing architecture for a relatively long period, a unique testing architecture and test method are applied to the 3GPP UE protocol tests.

5 ATS structure

The total TTCN specification for the UE testing is structured in a number of separate layered ATSs. The number of ATS being produced corresponds to the number of the 3GPP core specifications referred. The separation of ATSs reduces the size of ATSs. The layer-specific test preambles and test data can be confined to one test suite and parallel development of test suites can be facilitated. The separation of ATSs enables also easily to follow the evolution of the core specifications.

NAS ATSs
1) GSM MAP L3 ATS including MM, CC, GMM, SM test groups,
2) SMS ATS,

AS ATSs
1) RRC ATS including Idle mode test group,
2) RLC ATS,
3) MAC ATS
4) BMC ATS
5) PDCP ATS,
6) Intersystem ATS including the idle mode and handover test groups.

5.1 Modularity

The modular TTCN approach is used for the development of the 3GPP ATS specification work. Two modules, L2M and L3M are installed.

5.1.1 Working area structure

The working area is shown in Figure 1.

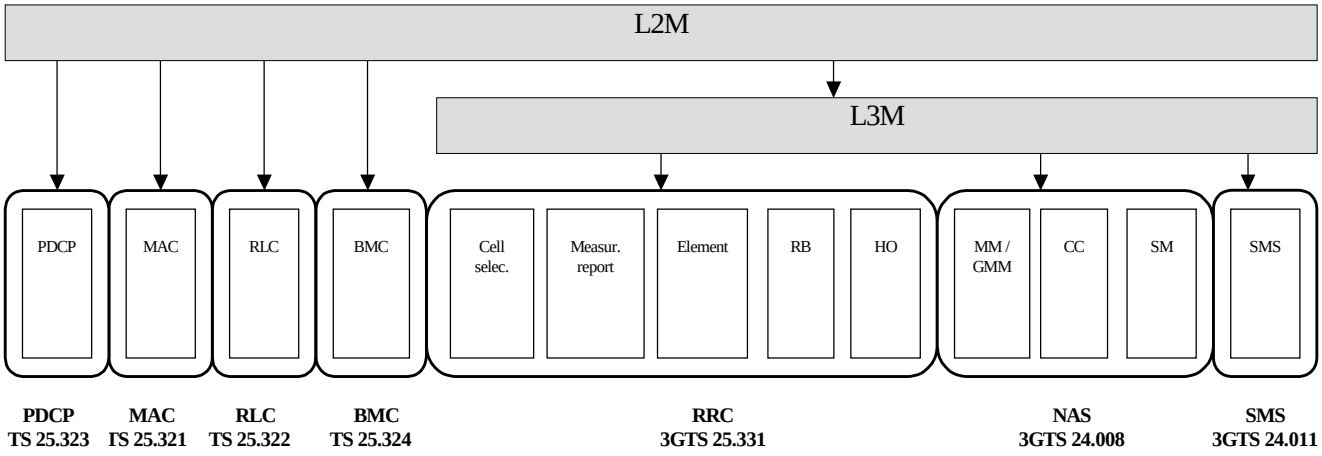

Figure 1: The proposed working area

The L2M (**L**ayer **2 M**odule) is a minimum module commonly for the layer 2 and layer 3 testing. The L3M (**L**ayer **3 M**odule) contains all the items to be shared by the RRC, NAS and SMS ATSs.

5.1.2 Contents of the modules

The L2M module includes objects related to the RRC, the layer 2 and the physical layer. It includes also all test steps needed by the layer 2 and layer 3 test cases for configurations and all objects related to the definition of the steps:

- Common test steps and default test steps defined as generic procedures in 3GPP TS 34.108[3];
- RRC declarations related to the steps: types, timers, PDU types, ASP type, PCOs, TSOs, constants;
- Related ICS and IXIT parameters needed for testing and respectively defined in 3GPP TS 34.123-2[2] and the present document;
- Defaults constraints based on the default message contents defined in 3GPP TS 34.108[3],
- MMI PCO and ASPs,
- All TTCN objects related to the SS configuration, e.g. PCOs, declaration of the components.

The L3M module includes the NAS configuration steps and all related TTCN objects:

- Common test steps and default test steps defined as generic procedures in 3GPP TS 34.108[3];
- NAS declarations related to these steps: types, PDU, ASP, PCOs, TSOs, constants;
- Related ICS and IXIT parameters needed for testing and respectively defined in 3GPP TS 34.123-2[2] and the present document;
- Default constraints based on the default message contents defined in 3GPP TS 34.108[3].

5.1.3 Example of a working platform

The figure 2 below shows the working platform for the user that is writing the RRC test cases.

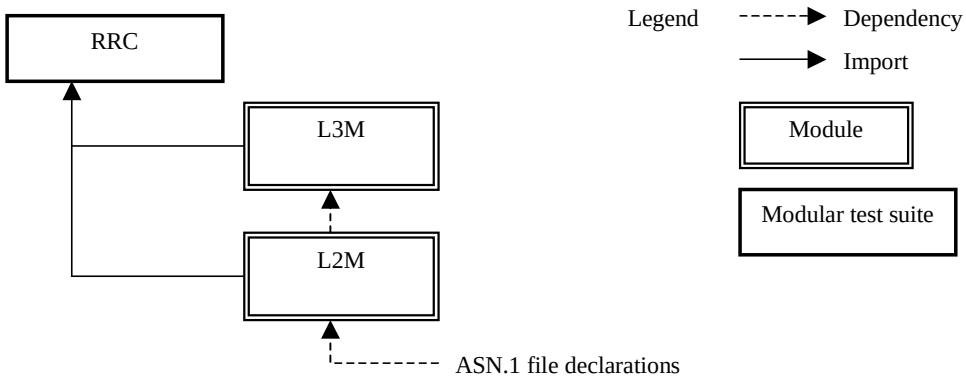

Figure 2: An example of working platform for RRC

6 Test method and Testing Architecture

6.1 Test method

The distributed single party test method is used for the UE testing. The lower tester configures the emulator and communicates with the UE under test via the emulator. An upper tester interfaces UE as (E)MMI..

All common parts in 3GPP TS 34.108 [3], 3GPP TS 34.109 [4] and 3GPP TS 34.123-2 [2] will be developed in a TTCN library including the declarations, default constraints, preambles and postambles. They have the following characteristics:

Very complex,

Worked in different layers,

Including data representing the radio parameters for SS setting and the data representing the UE capabilities (PICS parameters),

Including the generic procedures to bring the UE into certain test states or a test mode (C-plane),

Setting RABs at U-plane and SRBs in C-plane,

Being used by every test cases no matter which layer the test case belongs to,

No affect on the test verdict of PASS or FAIL.

The layer-specific test cases have the characteristics:

- - relatively simple and straight forward,
- - having narrow test scope and test purposes
- - test scenarios in a single layer (one PCO)
- - assigning the test verdict

6.2 Testing Architecture

A unique testing architecture is shown in the following figure.

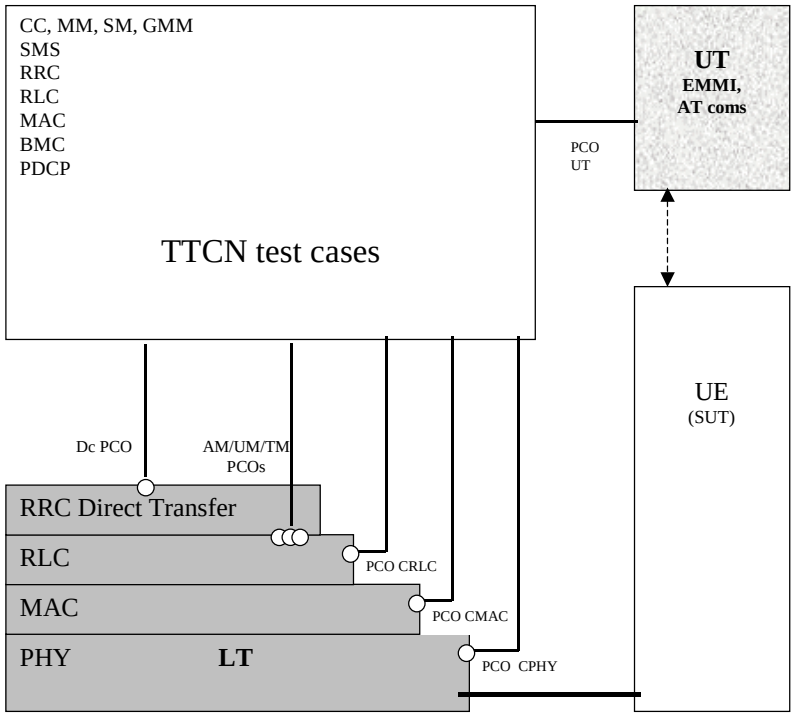

Figure 3: A unique testing architecture

6.2.1 Lower tester

The lower tester (LT) provides the test means for the execution of the test cases for CC, SM, MM, GMM, SMS, RRC, RLC, MAC, PDCP or BMC. The LT provides also the RLC, MAC and PHY emulators to communicate with the UE. The configuration and initialisation of the emulators are control by the TTCN via ASPs.

6.2.2 Configuration and initialisation

A number of TTCN test steps are designed for the generic setting.

1) Configuration of L1 of the tester, such as the cells, Physical channels and common transport channels via CPHY-PCO, configuration of MAC via CMAC-PCO and configuration of RLC layer via CRLC-PCO,

2) Sending system information via TR-PCO,

3) Establishment RRC connection via AM or UM-PCO,

4) Assigning a radio bearer via AM-PCO,

5) MM /GMM registration via Dc-PCO,

6) Establishment of a CS call or a PDP context via Dc-PCO,

7) Setting security parameters and control of integrity via CRLC- and ciphering via CRLC- and CMAC-PCO.

6.2.3 Upper tester

An upper tester (UT) exists in the test system. The UT interfaces toward UE with any optional EMMI (34.109, clause 7 [4]). TTCN communicates with the UT by passing coordination primitives via a Ut PCO. The primitives can either contain AT commands aiming at the automatic tests, or some informal commands as MMI, in order to request the UE for certain actions and to provide simple means for observations of UE.

6.2.4 TTCN

TTCN is used as specification language based on ETSI TR 101 666 [27] (TTCN 2++). The importation of ASN.1 modules and modular TTCN are two of the most important features used in the design of the ATSs.

The TTCN test suites have been designed to maximise the portability from the language TTCN 2 to TTCN 3.

6.2.5 Model extension

If a test case needs to handle a concurrent situation two or more LTs can be configured at the same time. The following test scenarios identified may require multiple testers in the test configuration.

1) Multiple cells HO tests,

2) Intersystem Idle mode and Handover tests.

6.2.6 Multiplexing of RLC services

For the RRC and NAS testing, the TTCN RRC test steps and the RRC emulator may share the same service access point (AM SAP). The RLC emulator shall provide separate message queues (buffers) for the TTCN RRC test stepsand the RRC emulator for the TTCN NAS test cases, according to the signalling radio bearer identities.

6.3 NAS test method and architecture

6.3.1 Test configuration

The NAS test method is shown in figure 4.

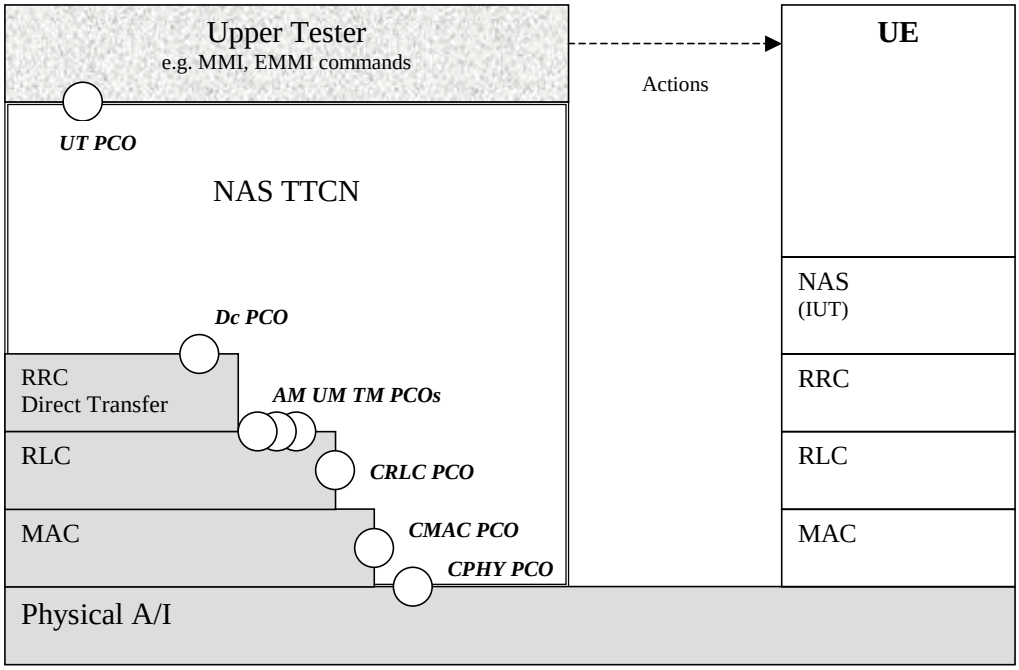

Figure 4: NAS testing architecture

The single layer distributed test method is used.

The Point of Control and Observation (PCO) are defined as the Dc (Dedicated control) SAP. The NAS test verdicts are assigned depending on the behaviours observed at the PCO.

The TTCN tester provides the NAS TTCN test cases and steps with a simple RRC direct transfer function which buffers the NAS PDU data, converts the data from the NAS TTCN table format into ASN.1, or in reverse way, and delivers all lower layer services of AM-SAP for RB3 and RB4.

The NAS TTCN test cases make also intensively use of the RRC TTCN test steps, in order to:

- Configure, initialise and control the L2 emulator,
- Initialise the UE for testing.

The RRC test steps, which are called by the NAS test cases or steps, interface with the RLC PCOs (UM, AM and TR), the control PCOs CRLC, CMAC and CPHY.

The General control (Gc) SAP and the Notification (Nt) SAP are not applied. Messages exchanged via these SAPs will be replaced with the corresponding RRC TTCN test steps.

The Ut PCO (so called logical interface [4]) is served as the interface to the UE EMMI to allow a remote control of operations, which have to be performed during execution of a test case such as to switch the UE on/off, initiate a call, etc.

6.4 RRC test method and architecture

6.4.1 Test configuration

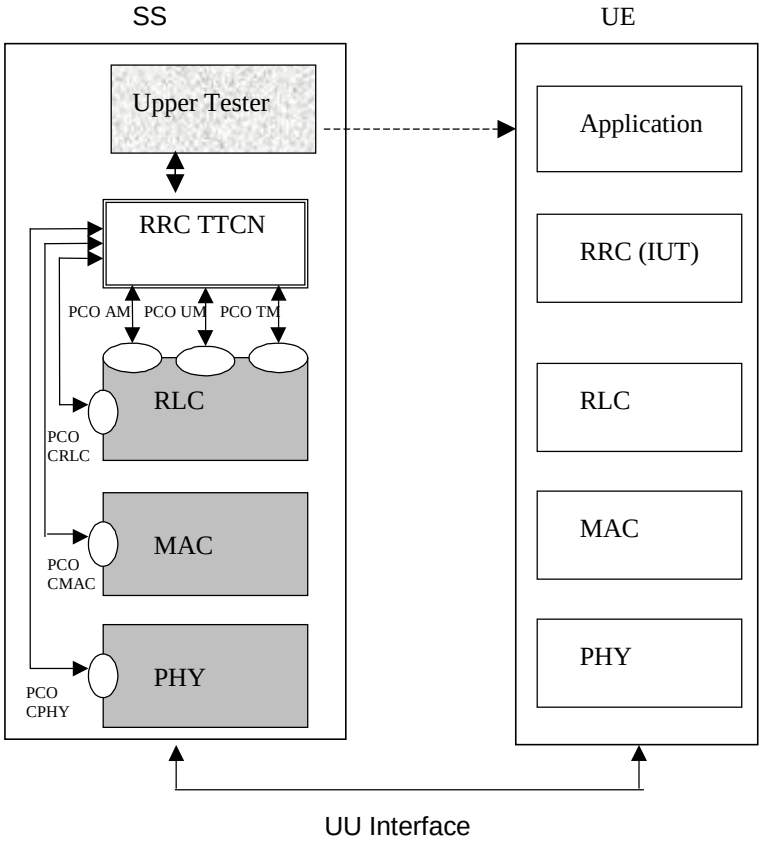

Figure 5: RRC testing architecture

The single layer distributed test method is used.

The PCOs are defined as the AM (Acknowledged Mode), UM (Unacknowledged Mode) and TM (Transparent Mode) SAPs. The RRC test verdicts are assigned depending on the behaviours observed at the PCO. The RRC TTCN interface also with the control PCOs CRLC, CMAC and CPHY, for the configuration, initialisation and control of the System Simulator.

The RRC TTCN test cases also make use of the NAS TTCN test steps in order to:

- Bring UE to Idle state,
- Bring UE to state U10

The NAS test steps, which are called by the RRC test cases or steps, interface with the Dc PCO.

The Ut PCO (so called logical interface [4]) is served as the interface to the UE EMMI to allow a remote control of operations, which have to be performed during execution of a test case such as to switch the UE on/off, initiate a call, etc.

6.5 RLC test method and architecture

6.5.1 Testing architecture

Figure 6 illustrates a typical realisation of the RLC ATS.

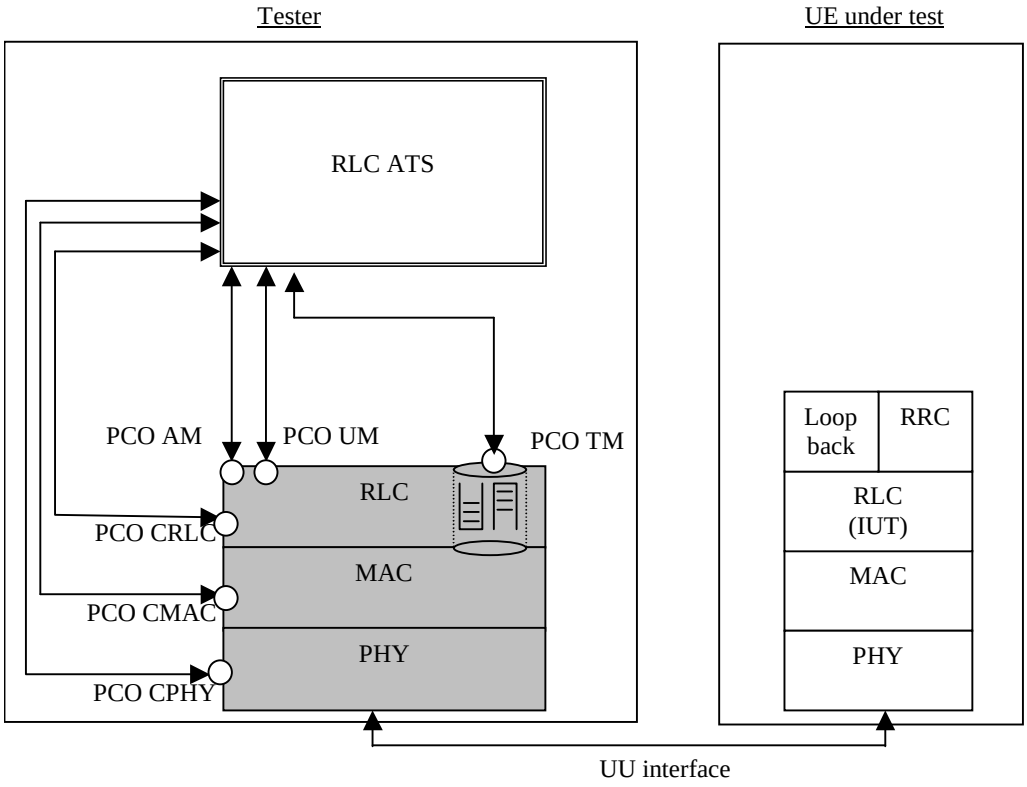

Figure 6: RLC ATS single party test method

The single party test method is used for RLC testing.

Separation of TTCN test cases from the configuration of the tester and initialisation of the UE is achieved by using test steps. For each RLC test case, common test steps will be used to perform the configuration of the tester and the appropriate generic setup procedures as described in 3GPP TS 34.108. These test steps will make use of PCOs AM, UM, TM, CRLC, CMAC, and CPHY.

Three PCOs are provided at the top of the RLC emulation in the tester, one corresponding to each of the available RLC modes: acknowledged, unacknowledged, and transparent. Routing information for different radio bearers used at these PCOs will be provided in ASP parameters.

The queues shown in the RLC emulation in figure 6 indicate that normal RLC transmit and receive buffering will be used to isolate the TTCN test suite from the real time issues involved if messages are sent directly to the MAC layer.

The RLC TTCN test cases make also use of the NAS TTCN test steps in order to bring UE to Idle state. The NAS test steps, which are called by the RLC test cases or steps, interface with the Dc PCO.

6.5.2 Test method

Figure 7 illustrates an example configuration for downlink UM testing. Uplink and AM tests will use similar configurations. A Tr-Entity is established on the tester side using a CRLC-CONFIG-REQ. A corresponding UM-Entity is created in the UE by sending a Radio Bearer Setup PDU. RLC PDUs are specified in the TTCN test suite, and sent to TM PCO. These PDUs shall be carefully designed so that the Tr-Entity will not perform any segmentation. To test reassembly in the UE side, the segmentation must be explicitly coded in TTCN. To test various aspects of the RLC header (e.g. sequence numbering, length indications etc), the RLC header must be explicitly coded in TTCN. Ciphering will not be tested using this approach, and will be disabled in the UE UM Entity.

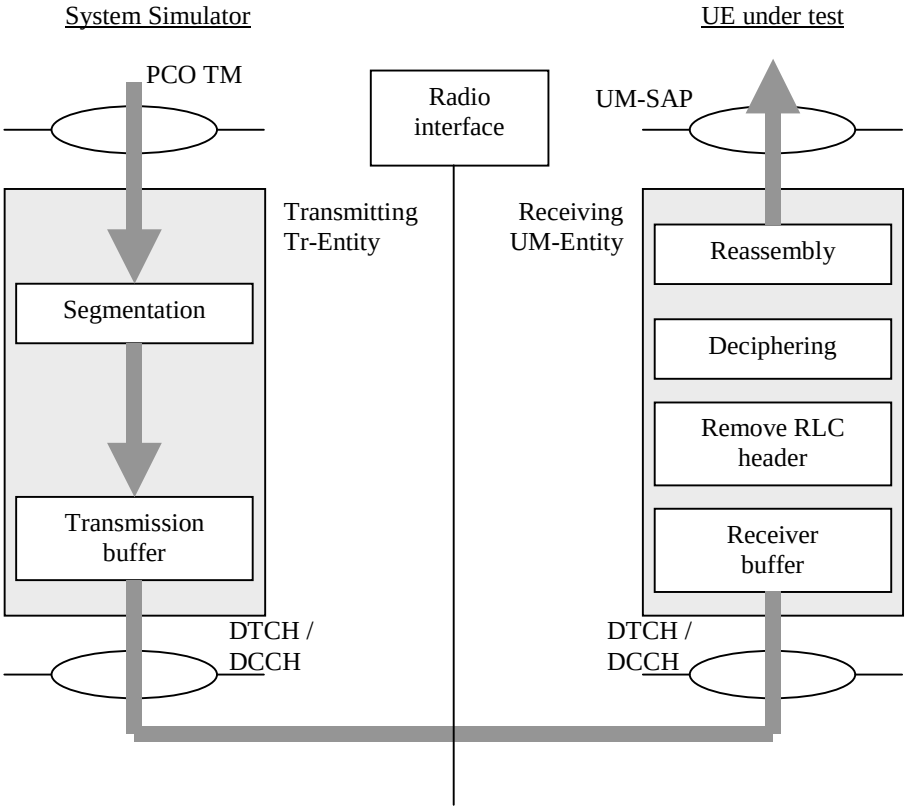

Figure 7: Example configuration for downlink RLC UM testing

All RLC tests that require uplink data will make use of the UE test loop mode 1 defined in 3GPP TS 34.109 [4]. The UE test loop mode 1 function provides all upper uester (UT) functionality required, so an Ut PCO is not required for RLC tests. Test Loop mode 1 is only available in the user plane, so all RLC tests will be performed in the user plane, using DTCH and DCCH logical channels mapped to DCH transport channels.

Ciphering tests will be performed using the PCOs UM and AM, with the UE test loop mode configured to ensure that uplink and downlink PUs are different sizes. Pseudo random data will be transmitted from the TTCN test case, ciphered by the SS RLC emulation UM or AM entity, and sent to the UE. The UE will then decipher, loop back (with different PU size), and cipher the data, and return it to the SS. The SS RLC UM or AM entity will then decipher the data, and pass it back to the TTCN, which will verify that the received data is identical to the transmitted data.

6.6 SMS test method and architecture

6.6.1 SMS CS test method and architecture

The test method used for SMS CS tests is the same as the NAS test method, see clause 6.3.

6.6.2 SMS PS test method and architecture

tbd

6.6.3 SMS Cell broadcasting test method and architecture

tbd

6.7 MAC test method and architecture

tbd

6.8 BMC test method and architecture

tbd

6.9 PDCP test

tbd

7 PCO and ASP definitions

7.1 NAS PCO and ASP definitions

7.1.1 NAS PCO Definitions

Table 1: Dc PCO Type Declarations

PCO Type Declarations	
PCO Type	Dc_SAP
Role	LT
Comments	The PCO type for NAS testing

Table 2: Dc PCO Declarations

PCO Declarations	
PCO Name	Dc
PCO Type	Dc_SAP
Role	LT
Comments	Carry transmission and reception of NAS messages

7.1.2 Primitives used at Dc PCO

The Dc PCO is used to transmit and receive NAS (MM, CC, SM, SS) messages. Two categories of primitives are operated at the Dc PCO:

RRC_DataReq for transmission of a NAS PDU;

RRC_DataInd for reception of a NAS PDU.

These primitives are declared in TTCN tabular form, see the following table.

Table 3: Primitives used at the Dc PCO

Primitive	Parameters	Use
RRC_DataInd	Cell identity RB identity LogicChGSM SapId CN domain id NAS message	The ASP is used to indicate the receipt of a NAS message using acknowledged operation
RRC_DataReq	Cell identity RB Identity LogicChGSM SapId CN domain id NAS message	The ASP is used to request the transmission of a NAS message using acknowledged operation.

The RB Identity and CN domain parameters defined in the primitives are mandatory for UTRAN and not applicable for GERAN.

The LogicChGSM and SapId parameters are mandatory for GERAN and not applicable for UTRAN. They are defined because they may be used for future TTCN test cases.

Except the initial, uplink and downlink direct transfer procedures, the NAS TTCN specification uses the TTCN test steps to realise all RRC functions for testing. The single layer test concept is kept for the NAS tests.

A simple RRC emulation shall be maintained for the NAS tests. It has four functions.

- Emulate the three direct transfer procedures,
- Convert the NAS downlink messages defined in 24.008 in table format to the NAS message in ASN.1 octet string specified in 25.331. Convert the NAS uplink message in the reverse way.
- PER encoding and decoding,
- Have the integrity protection.

RB3 and RB4 are specifically used for the NAS signalling. When an uplink message entered the receiving buffer at AM-SAP from the RLC emulation, either an RRC test step if running will take it out; or the RRC emulation if running will pick the received message from the buffer. Activation of any RRC test steps and activation of any NAS test steps at the same time shall be excluded in TTCN (no concurrency between them).

7.2 Ut PCO and ASP definitions

7.2.1 Ut PCO Declarations

The Ut PCO is served as the interface to the UE EMMI for remote control of operations, which have to be performed during execution of a test case such as to switch the UE on/off, initiate a call, etc.

Table 4: Declaration of the uppertester PCO type

PCO Type Declarations	
PCO Type	MMI
Role	UT
Comments	The PCO type for MMI or EMMI of the upper tester

Table 5: Declaration of the Ut PCO

PCO Declarations	
PCO Name	Ut
PCO Type	MMI
Role	UT
Comments	Carry transmission commands and reception of results for the upper tester

7.2.2 Primitives used at Ut PCO

The Ut PCO is used to indicate to the upper tester actions and to receive the acknowledgement of these actions. The AT commands are used wherever the suitable commands exist whithin 3GPP TS 27.007 [23], 3GPP TS 27.005 [22] and 3GPP TS 27 060 [24]. An MMI command is used, when AT commands does not exit for the action to performed. The primitives used at the Ut PCO, are declared in TTCN tabular form, see the following table.

Table 6: Primitives used at the Ut PCO

Primitive	Parameters	Use
AT_CmdReq	Command: IA5String SMS_BlockMode: HEXSTRING	Request an AT command to the upper tester.
AT_CmdInd	Command: IA5String SMS_BlockMode: HEXSTRING	Indication of a result from the upper tester.
AT_CmdCnf	Result: BOOLEAN ResultString: IA5String SMS_BlockMode: HEXSTRING	Return a positive or negative result from the command previously sent. The String parameter is optional.
MMI_CmdReq	Command: IA5String	Request a command to the upper tester.
MMI_CmdCnf	Result: BOOLEAN ResultString: IA5String	Return a positive or negative result from the command previously sent. The String parameter is optional.

The AT_CmdReq primitive for sending AT commands is mostly used to trigger electronically an uplink access, such as initiating of a call, attaching or detaching, starting packet data transfer etc. The MMI_ primitive is defined mainly for observation of some test events via a test operator, such as checking DTMF tone or checking called party number, etc.

The AT_CmdInd primitive for receiving AT commands is mostly used to transfer unsolicited result codes from the UE to the lower tester.

The SMS_BlockMode parameter is used to control and observe the Block mode procedure for SMS. This parameter is not yet used; it is defined for future development. The Command and SMS_BlockMode parameters are mutually exclusive

For the Command in the AT_CmdReq and AT_CmdInd primitives, the verbose format is used as defined in 3GPP TS 27.007. For the Command in MMI_CmdReq, just a descriptive IA5 string line, like "Check DTMF tone" is used.

7.3 RRC PCO and ASP definitions

7.3.1 AM/UM/TM PCO and ASP definitions

7.3.1.1 SAP and PCO for data transmission and reception

Table 7: Declaration of the RRC PCO Type

PCO Type Definition	
PCO Type	DSAP
Role	LT
Comment	DATA transmission and reception

Table 8: PCO TM declaration

PCO Type Definition	
PCO Name	TM
PCO Type	DSAP
Role	LT
Comment	Carry Transparent Mode RLC PDU

Table 9: PCO AM declaration

PCO Type Definition	
PCO Name	AM
PCO Type	DSAP
Role	LT
Comment	Carry Acknowledged Mode RLC PDU

Table 10: PCO UM declaration

PCO Type Definition	
PCO Name	UM
PCO Type	DSAP
Role	LT
Comment	Carry Unacknowledged Mode RLC PDU

7.3.2 Control PCO and ASP

7.3.2.1 SAP and PCO for control primitives transmission and reception

Table 11: SAP declaration

PCO Type Definition	
PCO Type	CSAP
Role	LT
Comment	Control primitives transmission and reception

Table 12: PCO CPHY

PCO Definition	
PCO Name	CPHY
PCO Type	CSAP
Role	LT
Comment	Control Physical Layer

Table 13: PCO CRLC

PCO Type Definition	
PCO Name	CRLC
PCO Type	CSAP
Role	LT
Comment	Control RLC Layer

Table 14: PCO CMAC

PCO Type Definition	
PCO Name	CMAC
PCO Type	CSAP
Role	LT
Comment	Control MAC Layer

7.3.2.2 Control ASP Type Definition

7.3.2.2.1 CPHY_TrCH_Config_REQ

ASN.1 ASP Type Definition	
Type Name	CPHY_TrCH_Config_REQ
PCO Type	CSAP
Comment	To request to configure the transport channel
Type Definition	

```
SEQUENCE     {
               cellId                    INTEGER(0..15),
               routingInfo               RoutingInfo,
               ratType                   RatType,
               configMessage             CphyTrchConfigReq
}
```

ASN.1 Type Definition	
Type Name	CphyTrchConfigReq
Comment	To request to configure the transport channel
Type Definition	

```
SEQUENCE     {
             ulconnectedTrCHList      SEQUENCE (SIZE (0..maxTrCH)) OF SEQUENCE {
               trchid                   TransportChannelIdentity,
                TransportChannelInfo     CHOICE   {
               ul_CommonTrCHInfo        CommonTransChTFS,
               ul_DedTrCHInfo           DedicatedTransChTFS
               },
               }        OPTIONAL,
               ulctfcList               SEQUENCE (SIZE (0..maxTFC)) OF SEQUENCE {
               ctfc                     INTEGER (0..1023),
               gainFactorInformation    PowerOffsetInformation
}        OPTIONAL,
               dlconnectedTrCHList      SEQUENCE (SIZE (0..maxTrCH)) OF SEQUENCE {
               trchid                   TransportChannelIdentity,
                TransportChannelInfo     CHOICE   {
               dl_CommonTrCHInfo        CommonTransChTFS,
               dl_DedTrCHInfo           DedicatedTransChTFS
               }
               }        OPTIONAL,
               dlctfcList               SEQUENCE (SIZE (0..maxTFC)) OF SEQUENCE {
               ctfc                     INTEGER (0..1023)
}        OPTIONAL
}
```

ASN.1 Type Definition	
Type Name	RoutingInfo
Comment	To route between each channels.
Type Definition	

```
CHOICE     {
        PhysicalChannelIdentity     INTEGER   {0..31},
        transportChannelIdentity    TransportChannelIdentity,
        logicalChannelIdentity      LogicalChannelIdentity,
        rB_Identity                 RB_Identity,
        cn-DomainIdentity           CN-DomainIdentity
}
```

ASN.1 Type Definition	
Type Name	RatType
Comment	To select route between each channels.
Type Definition	

```
ENUMERATED     {
                fdd,tdd
}
```

7.3.2.2.2 CPHY_RL_Setup_REQ

ASN.1 ASP Type Definition	
Type Name	CPHY_RL_Setup_REQ
PCO Type	CSAP
Comment	To request to setup the Radio Link
Type Definition	

```
SEQUENCE     {
            cellId                      INTEGER(0..15),
            routingInfo                 RoutingInfo,
            ratType                     RatType,
            setupMessage                CphyRlSetupReq
}
```

ASN.1 Type Definition	
Type Name	CphyRlSetupReq
Comment	To request to setup the Radio Link
Type Definition	

```
SEQUENCE    {
        physicalChannelInfo         CHOICE    {
        primaryCPICHInfo            PrimaryCPICHInfo,
        secondaryCPICHInfo          SecondaryCPICHInfo,
        primarySCHInfo              PrimarySCHInfo,
        secondarySCHInfo            SecondarySCHInfo,
        primaryCCPCHInfo            PrimaryCCPCHInfo,
        secondaryCCPCHInfo          SecondaryCCPCHInfo,
        pRACHInfo                   PRACHInfo,
        pICHInfo                    PICHInfo,
        aICHInfo                    AICHInfo,
        dPCHInfo                    DPCHInfo
     -- pCPCHInfo                   PCPCHInfo,
    --  aP_ICHInfo                  AP_AICHInfo,
     -- cD_ICHInfo                  CD_ICHInfo,
     -- cD_CA_ichInfo               CD_CA_ICHInfo,
    --  cSICHInfo                   CSICHInfo,
    --  pDSCHInfo                   PDSCHInfo,
    --  pUSCHInfo                   PUSCHinfo
    --  }
}
```

ASN.1 Type Definition	
Type Name	PrimaryCPICHInfo
Comment	
Type Definition	

```
SEQUENCE {
            dl_TxPower_PCPICH       DL_TxPower_PCPICH,
            tx_diversityIndicator   BOOLEAN
}
```

ASN.1 Type Definition	
Type Name	SecondaryCPICHInfo
Comment	

Type Definition

```
SEQUENCE {
            scramblingCode            INTEGER{0..15},
            dl_ChannelizationCode     SF512_AndCodeNumber,
            dl_TxPower                DL_TxPower
}
```

ASN.1 Type Definition	
Type Name	PrimarySCHInfo
Comment	

Type Definition

```
SEQUENCE {
            tstdIndicator             BOOLEAN,
            dl_TxPower                DL_TxPower
}
```

ASN.1 Type Definition	
Type Name	SecondarySCHInfo
Comment	

Type Definition

```
SEQUENCE {
            tstdIndicator          BOOLEAN,
            dl_TxPower             DL_TxPower
}
```

ASN.1 Type Definition	
Type Name	PrimaryCCPCHInfo
Comment	

Type Definition

```
SEQUENCE {
           sttd_Indicator          BOOLEAN,
           dl_TxPower              DL_TxPower
        --  timeSlot                TimeSlot                OPTIONAL,
        --  burstType               BurstType               OPTIONAL,
        --  offset                  Offset                  OPTIONAL,
        --  repetitionPeriod        RepetitionPeriod        OPTIONAL,
        --  repetitionLength        RepetitionLength        OPTIONAL,
}
```

ASN.1 Type Definition	
Type Name	SecondaryCCPCHInfo
Comment	

Type Definition

```
SEQUENCE {
            scramblingCode           INTEGER(0..15),
            dl_ChannelizationCode    SF512_AndCodeNumber,
            sCCPCHSlotFormat         SCCPCHSlotFormat,
            timingOffset             INTEGER (0..149),
            positionFixedOrFlexible  PositionFixedOrFlexible,
            sttd_Indicator            BOOLEAN,
            dl_TxPower                DL_TxPower,
            powerOffsetOfTFCI_PO1    INTEGER (0..50),
            powerOffsetOfPILOT_PO3   INTEGER (0..50)
            --  timeSlot                 TimeSlot                  OPTIONAL,
             -- burstType                BurstType                 OPTIONAL,
             -- midambleShift            MidambleShift             OPTIONAL,
             -- offset                   Offset                    OPTIONAL,
             -- repetitionPeriod         RepetitionPeriod          OPTIONAL,
             -- repetitionLength         RepetitionLength          OPTIONAL,
             -- tFCIPresence             TFCIPresence              OPTIONAL,
}
```

ASN.1 Type Definition	
Type Name	PRACHInfo
Comment	

Type Definition

```
SEQUENCE {
            fdd_tdd                         CHOICE  {
            fdd                             SEQUENCE   {
            preambleSignature               AvailableSignatures,
            spreadingFactorForDataPart      SF_PRACH,
            preambleScramblingCode           PreambleScramblingCodeWordNumber,
            puncturingLimit                 PuncturingLimit,
            accessSlot                      AvailableSubChannelNumbers
            },
            tdd                             SEQUENCE   {
            -- timeSlot                  TimeSlot,
            -- spreadingCode             SpreadingCode,
            -- midambleCode              MidambleCode,
            }
}
}
```

ASN.1 Type Definition	
Type Name	PICHInfo
Comment	

Type Definition

```
SEQUENCE {
            pichinfo                PICH_Info,
            dl_TxPower              DL_TxPower
}
```

ASN.1 Type Definition	
Type Name	AICHInfo
Comment	

Type Definition

```
SEQUENCE {
            aichinfo                AICH_Info,
            dl_TxPower              DL_TxPower
}
```

ASN.1 Type Definition	
Type Name	DPCHInfo
Comment	At least one of the fields shall be present.

Type Definition

```
SEQUENCE {
            ul_DPCH_Info              UL_DPCH_Info   OPTIONAL,
            dl_DPCHInfo               DL_DPCHInfo   OPTIONAL
}
```

ASN.1 Type Definition	
Type Name	DL_DPCHInfo
Comment	
Type Definition	

```
SEQUENCE {
            dl_CommonInformation        DL_CommonInformation,
            dl_DPCH_InfoPerRL           DL_DPCH_InfoPerRL,
            powerOffsetOfTFCI_PO1       INTEGER (0..50),
            powerOffsetOfTPC_PO2        INTEGER (0..50),
            powerOffsetOfPILOT_PO3      INTEGER (0..50),
            dl_TxPower                  DL_TxPower,
            dl_TxPowerMax               DL_TxPower,
            dl_TxPowerMin               DL_TxPower
}
```

ASN.1 Type Definition	
Type Name	DL_TxPower_PCPICH
Comment	Absolute Tx Power of PCPICH
Type Definition	

```
INTEGER (-60..-30)
```

ASN.1 Type Definition	
Type Name	DL_TxPower
Comment	Relative Tx Power for PCPICH
Type Definition	

```
INTEGER (-35..+15)
```

ASN.1 Type Definition	
Type Name	SCCPCHSlotFormat
Comment	Reference to TS25.211
Type Definition	

```
INTEGER (0..17)
```

ASN.1 Type Definition	
Type Name	UL_DPCCHSlotFormat
Comment	Reference to TS25.211
Type Definition	

```
INTEGER (0..5)
```

7.3.2.2.3 CPHY_RL_Modify_REQ

ASN.1 ASP Type Definition	
Type Name	CPHY_RL_Modify_REQ
PCO Type	CSAP
Comment	To request to modify the Radio Link HardHandover (PhysicalChannelReconfig) ChannelisationCodeChange FrequencyChange PhysicalChannelModifyForTrCHReconfig CompressedMode(PhysicalChannelReconfig) Re_Synchronized HardHandover Softhandover

Type Definition

```
SEQUENCE     {
               cellId                    INTEGER(0..15),
               routingInfo               RoutingInfo,
               ratType                   RatType,
               modifyMessage             CphyRlModifyReq
}
```

ASN.1 Type Definition	
Type Name	CphyRlModifyReq
Comment	

Type Definition

```
SEQUENCE    {
        branchInfo                  BranchInfo,
        PhysicalChannelInfo         CHOICE    {
        CompessedModeSetup          CompressedModeSetup,
        secondaryCCPCHInfo          SecondaryCCPCHInfo,
        pRACHInfo                   PRACHInfo,
        dPCHInfo                    DPCHInfo,
        }             }
```

ASN.1 Type Definition	
Type Name	CompressedModeSetup
Comment	

Type Definition

```
SEQUENCE  {
         ul_CompressedModeMethod          UL_CompressedModeMethod         OPTIONAL,
         dl_CompressedModeMethod          DL_CompressedModeMethod         OPTIONAL,
         dpch_CompressedModeInfo          DPCH_CompressedModeInfo
}
```

ASN.1 Type Definition	
Type Name	BranchInfo
Comment	

Type Definition

```
ENUMERATED  {
            NotChange, Add, Delete
}
```

7.3.2.2.4 CPHY_Commit_REQ

ASN.1 ASP Type Definition	
Type Name	CPHY_Commit_REQ
PCO Type	CSAP
Comment	To request the timing to modify the Radio Link
Type Definition	

```
SEQUENCE    {
        cellId                  INTEGER(0..15),
        routingInfo             RoutingInfo,
        activationTime          INTEGER (0..255)
}
```

7.3.2.2.5 CPHY_TrCH_Release_REQ

ASN.1 ASP Type Definition	
Type Name	CPHY_TrCH_Release_REQ
PCO Type	CSAP
Comment	To request to release the Radio Link
Type Definition	

```
SEQUENCE    {
        cellId                  INTEGER(0..15),
        routingInfo             RoutingInfo
}
```

7.3.2.2.6 CPHY_RL_Release_REQ

ASN.1 ASP Type Definition	
Type Name	CPHY_RL_Release_REQ
PCO Type	CSAP
Comment	To request to release the Radio Link
Type Definition	

```
SEQUENCE    {
        cellId                  INTEGER(0..15),
        routingInfo             RoutingInfo
}
```

7.3.2.2.7 CPHY_RL_Release_CNF

ASN.1 ASP Type Definition	
Type Name	CPHY_RL_Release_CNF
PCO Type	CSAP
Comment	PHY emulator confirms that a specified physical channel has been released.
Type Definition	

```
SEQUENCE    {
        cellId                  INTEGER(0..15),
        routingInfo             RoutingInfo,
}
```

7.3.2.2.8 CRLC_Config_REQ

ASN.1 ASP Type Definition	
Type Name	CRLC_Config_REQ
PCO Type	CSAP
Comment	To request to setup, reconfigure or release RLC entity
Type Definition	

```
SEQUENCE     {
             cellId                    INTEGER(0..15),
             routingInfo               RoutingInfo,
             ratType                   RatType,
             configMessage             CrlcConfigReq
}
```

ASN.1 Type Definition	
Type Name	CrlcConfigReq
Comment	To request to setup, re_configure or release RLC entity
Type Definition	

```
CHOICE {
             setup                     RBInfo,
             reconfigure               RBInfo,
             release                   NULL
}
```

ASN.1 Type Definition	
Type Name	RBInfo
Comment	
Type Definition	

```
SEQUENCE (
              pdcp_Info                PDCP_Info           OPTIONAL,
              sS_rlc_Info              SS_RLC_Info         OPTIONAL,
              rB_LogCH_Mapping         RB_LogCH_Mapping
}
```

ASN.1 Type Definition	
Type Name	RB_LogCH_Mapping
Comment	Provide mapping information between RB, logical channel and CN domain.
Type Definition	

```
SEQUENCE {
    uLlogicalChannelIdentity              LogicalChannelIdentity  OPTIONAL,
    dLlogicalChannelIdentity              LogicalChannelIdentity  OPTIONAL,
    cn-DomainIdentity                     CN-DomainIdentity       OPTIONAL
}
```

ASN.1 Type Definition	
Type Name	SS_RLCInfo
Comment	
Type Definition	

```
SEQUENCE {
             sS_ul_RLC_Mode          DL_RLC_Mode         OPTIONAL,
             sS_dl_RLC_Mode          SS_DL_RLC_Mode          OPTIONAL
}
```

ASN.1 Type Definition	
Type Name	SS_DL_RLC_Mode
Comment	
Type Definition	

```
SEQUENCE {
             dl_PayloadSize                      PayloadSize     OPTIONAL,
             dl_RLCModeInfo                      UL_RLC_Mode
}
```

ASN.1 Type Definition	
Type Name	PayloadSize
Comment	
Type Definition	
INTEGER (0..4992)	

7.3.2.2.9 CRLC_Config_CNF

ASN.1 ASP Type Definition	
Type Name	CRLC_Config_CNF
PCO Type	CSAP
Comment	For RLC emulator to confirm that a previous attempt to establish, re_configure or release a radio bearer has been successful.
Type Definition	

```
SEQUENCE  {
            cellId                  INTEGER(0..15),
            routingInfo             RoutingInfo
}
```

7.3.2.2.10 CRLC_Suspend_REQ

ASN.1 ASP Type Definition	
Type Name	CRLC_Suspend_REQ
PCO Type	CSAP
Comment	To request to suspend data transmission
Type Definition	

```
SEQUENCE {
            cellId                  INTEGER(0..15),
            routingInfo             RoutingInfo,
            n                       RLC_SequenceNumber
}
```

7.3.2.2.11 CRLC_Suspend_CNF

ASN.1 ASP Type Definition	
Type Name	CRLC_Suspend_CNF
PCO Type	CSAP
Comment	To confirm to suspend data transmission
Type Definition	

```
SEQUENCE {
            cellId                  INTEGER(0..15),
            routingInfo             RoutingInfo,
            sn                      RLC_SequenceNumber
}
```

7.3.2.2.12 CRLC_Resume_REQ

ASN.1 ASP Type Definition	
Type Name	CRLC_Resume_REQ
PCO Type	CSAP
Comment	To request to resume data transmission
Type Definition	

```
SEQUENCE  {
            cellId                  INTEGER(0..15),
            routingInfo             RoutingInfo
}
```

7.3.2.2.13 CMAC_Config_REQ

ASN.1 ASP Type Definition	
Type Name	CMAC_Config_REQ
PCO Type	CSAP
Comment	To request to configure MAC entity
Type Definition	

```
SEQUENCE     {
            cellId                      INTEGER(0..15),
            routingInfo                 RoutingInfo,
            ratType                     RatType,
            configMessage               CmacConfigReq
}
```

ASN.1 Type Definition	
Type Name	CmacConfigReq
Comment	To request to configure MAC
Type Definition	

```
SEQUENCE {
             uE_Info                                UE_Info,
             trCHInfo                               TrCHInfo,
             trCH_LogCHMapping                      TrCH_LogCHMappingList1
             -- RACHTrasmissionCtrolElements          TBD,
             -- CPCHTransmissionControlElements       TBD
}
```

ASN.1 Type Definition	
Type Name	UE_Info
Comment	
Type Definition	

```
SEQUENCE {
        u_RNTI            SEQUENCE  {
        srnc_Identity     BIT STRING (SIZE (12)),
        s_RNTI            BIT STRING (SIZE (20))
        }                 OPTIONAL,
        c_RNTI            BIT STRING (SIZE (16))              OPTIONAL
}
```

ASN.1 Type Definition	
Type Name	TrCH_LogCHMappingList1
Comment	
Type Definition	

```
SEQUENCE   {
        UlconnectedTrCHList     SEQUENCE (SIZE (1..maxulTrCH)) OF SEQUENCE {
                trchid                  TransportChannelIdentity,
                trCH_LogCHMappingList   TrCH_LogCHMappingList
                                },   OPTIONAL,
        dlconnectedTrCHList     SEQUENCE (SIZE (1..maxdlTrCH)) OF SEQUENCE {
                trchid                  TransportChannelIdentity,
                trCH_LogCHMappingList   TrCH_LogCHMappingList
                                },   OPTIONAL
}
```

ASN.1 Type Definition	
Type Name	TrCH_LogCHMappingList
Comment	
Type Definition	

```
SEQUENCE (SIZE (1..maxLogCHperTrCH)) OF TrCH_LogicalChannelMapping
```

ASN.1 Type Definition	
Type Name	TrCHInfo
Comment	

Type Definition

```
SEQUENCE   {
    UlconnectedTrCHList         SEQUENCE (SIZE (1..maxulTrCH)) OF SEQUENCE {
                trchid                  TransportChannelIdentity,
                TransportChannelInfo    CHOICE  {
                    ul_CommonTrCHInfo       CommonTransChTFS,
                    ul_DedTrCHInfo          DedicatedTransChTFS
                                                },
                                        }       OPTIONAL,
                ulctfcList      SEQUENCE (SIZE (0..maxTFC)) OF SEQUENCE {
                    ctfc                    INTEGER (0..1023),
                    }           OPTIONAL,
    dlconnectedTrCHList         SEQUENCE (SIZE (1..maxdlTrCH)) OF SEQUENCE {
               trchid                   TransportChannelIdentity,
               TransportChannelInfo     CHOICE   {
               dl_CommonTrCHInfo        CommonTransChTFS,
               dl_DedTrCHInfo           DedicatedTransChTFS
               },
               }        OPTIONAL,
               dlctfcList               SEQUENCE (SIZE (0..maxTFC)) OF SEQUENCE {
               ctfc                     INTEGER (0..1023)
               }        OPTIONAL
}
```

ASN.1 Type Definition	
Type Name	TrCH_LogicalChannelMapping
Comment	

Type Definition

```
SEQUENCE    {
            CHOICE  {
                ul_LogicalChannelMapping            SS_UL_LogicalChannel_Mapping,
                dl_LogicalChannelMapping            SS_DL_LogicalChannel_Mapping
                    },
            rB_Identity                 RB_Identity,
            cn-DomainIdentity           CN-DomainIdentity   OPTIONAL
}
```

ASN.1 Type Definition	
Type Name	SS_UL_LogicalChannel_Mapping
Comment	

Type Definition

```
SEQUENCE {
    ul_TransportChannelType             SS_UL_TransportChannel_Type,
    logicalChannelIdentity              LogicalChannelIdentity
}
```

ASN.1 Type Definition	
Type Name	SS_DL_LogicalChannel_Mapping
Comment	

Type Definition

```
SEQUENCE {
    dlTransportChannelType              SS_DL_TransportChannel_Type,
    logicalChannelIdentity              LogicalChannelIdentity,
    mac_LogicalChannelPriority          MAC_LogicalChannelPriority  OPTIONAL
}
```

ASN.1 Type Definition	
Type Name	SS_UL_TransportChannel_Type
Comment	
Type Definition	

```
ENUMELATED {
    dch,
    rach,
    cpch,
    usch
}
```

ASN.1 Type Definition	
Type Name	MAC_LogicalChannelPriority
Comment	
Type Definition	

```
INTEGER (1..8)
```

ASN.1 Type Definition	
Type Name	SS_DL_TransportChannel_Type
Comment	
Type Definition	

```
ENUMELATED {
    dch,
    fach,
    bch,
    pch,
    dsch
}
```

7.3.2.2.14 CMAC_Config_CNF

ASN.1 ASP Type Definition	
Type Name	CMAC_Config_CNF
PCO Type	CSAP
Comment	For MAC emulator to report that a previous attempt to setup, reconfigure or release a logical channel is successful.
Type Definition	

```
SEQUENCE {
            cellId                  INTEGER(0..15),
            routingInfo             RoutingInfo
}
```

7.3.2.2.15 RLC_TR_DATA_REQ

ASN.1 ASP Type Definition	
Type Name	RLC_TR_DATA_REQ
PCO Type	CSAP
Comment	To request to transmit DATA using transparent mode.
Type Definition	

```
SEQUENCE {
            cellId                  INTEGER(0..15),
            routingInfo             RoutingInfo,
                tM_Message     Message
}
```

7.3.2.2.16 RLC_TR_DATA_IND

ASN.1 ASP Type Definition	
Type Name	RLC_TR_DATA_IND
PCO Type	DSAP
Comment	To indicate to receive DATA using transparent mode.
Type Definition	

```
SEQUENCE {
            cellId                  INTEGER(0..15),
            routingInfo             RoutingInfo,
            tM_Message              Message
}
```

7.3.2.2.17 RLC_AM_DATA_REQ

ASN.1 ASP Type Definition	
Type Name	RLC_AM_DATA_REQ
PCO Type	DSAP
Comment	To request to transmit DATA using acknowledged mode.
Type Definition	

```
SEQUENCE {
            CellId                  INTEGER(0..15),
            RoutingInfo             RoutingInfo,
                aM_Message          Message,
            mui                     INTEGER(0..4095)   OPTIONAL
}
```

7.3.2.2.18 RLC_AM_DATA_IND

ASN.1 ASP Type Definition	
Type Name	RLC_AM_DATA_IND
PCO Type	DSAP
Comment	To indicate to receive DATA using acknowledged mode.
Type Definition	

```
SEQUENCE {
            cellId                  INTEGER(0..15),
            routingInfo             RoutingInfo,
            aM_Message              Message,
            integrityResult         IntegrityResult     OPTIONAL
}
```

ASN.1 Type Definition	
Type Name	IntegrityResult
Comment	
Type Definition	

```
ENUMERATED {
            i_pass(0), i_fail(1)
}
```

7.3.2.2.19 RLC_AM_DATA_CNF

ASN.1 ASP Type Definition	
Type Name	RLC_AM_DATA_CNF
PCO Type	DSAP
Comment	For RLC emulator to report to the upper layer that a previously transmitted SDU has been acknowledged correctly by the UE

Type Definition

```
SEQUENCE {
            cellId                  INTEGER(0..15),
            routingInfo             RoutingInfo,
            mui                     INTEGER(0..4095)
}
```

ASN.1 PDU Type Definition	
Type Name	Message
PCO Type	DSAP
Comment	

Type Definition

```
CHOICE   {
         dL_DCCH_Message                DL_DCCH_Message,
         uL_DCCH_Message                UL_DCCH_Message,
         dL_CCCH_Message                DL_CCCH_Message,
         uL_CCCH_Message                UL_CCCH_Message,
         pCCH_Message                   PCCH_Message,
         dL_SHCCH_Message               DL_SHCCH_Message,
         uL_SHCCH_Message               UL_SHCCH_Message,
         bCCH_FACH_Message              BCCH_FACH_Message,
         bCCH_BCH_Message               BCCH_BCH_Message
}
```

7.3.2.2.20 RLC_UM_DATA_REQ

ASN.1 ASP Type Definition	
Type Name	RLC_UM_DATA_REQ
PCO Type	DSAP
Comment	To request to transmit DATA using unacknowledged mode.

Type Definition

```
SEQUENCE {
            cellId                  INTEGER(0..15),
            routingInfo             RoutingInfo,
                uM_Message            Message
}
```

7.3.2.2.21 RLC_UM_DATA_IND

ASN.1 ASP Type Definition	
Type Name	RLC_UM_DATA_IND
PCO Type	DSAP
Comment	To indicate to receive DATA using unacknowledged mode.

Type Definition

```
SEQUENCE {
            cellId                  INTEGER(0..15),
            routingInfo             RoutingInfo,
            uM_Message              Message,
            integrityResult         IntegrityResult      OPTIONAL
}
```

7.3.2.2.22 CPHY_TrCH_Config_CNF

ASN.1 ASP Type Definition	
Type Name	CPHY_TrCH_Config_CNF
PCO Type	CSAP
Comment	To confirm to configure the transport channel
Type Definition	

```
SEQUENCE     {
             cellId                  INTEGER(0..15),
             routingInfo              RoutingInfo
}
```

7.3.2.2.23 CPHY_RL_Setup_CNF

ASN.1 ASP Type Definition	
Type Name	CPHY_RL_Setup_CNF
PCO Type	CSAP
Comment	To confirm to setup the Radio Link
Type Definition	

```
SEQUENCE    {
             cellId                  INTEGER(0..15),
             routingInfo              RoutingInfo
}
```

7.3.2.2.24 CPHY_RL_Modify_CNF

ASN.1 ASP Type Definition	
Type Name	CPHY_RL_Modify_CNF
PCO Type	CSAP
Comment	To confirm to modify the Radio Link
Type Definition	

```
SEQUENCE    {
             cellId                  INTEGER(0..15),
             routingInfo              RoutingInfo
}
```

7.3.2.2.25 CPHY_TrCH_Release_CNF

ASN.1 ASP Type Definition	
Type Name	CPHY_TrCH_Release_CNF
PCO Type	CSAP
Comment	To confirm to release the Radio Link
Type Definition	

```
SEQUENCE    {
             cellId                  INTEGER(0..15),
             routingInfo              RoutingInfo
}
```

7.3.2.2.26 CRLC_STATUS_IND

ASN.1 ASP Type Definition	
Type Name	CRLC_Status_IND
PCO Type	CSAP
Comment	To report the occurrence of certain events to RRC. Note: the possible event types to be defined for this ASP is FFS.

Type Definition

```
SEQUENCE     {
             cellId                    INTEGER(0..15),
             routingInfo               RoutingInfo,
             ratType                   RatType,
             statusInd                 CrlcStatusInd
}
```

ASN.1 Type Definition	
Type Name	CrlcStatusInd
Comment	

Type Definition

```
ENUMERATED
{
          DataLinkFailure,
          MaxRESET,
          SDUDiscarded
          -- __ More event types are to be added here
}
```

7.3.2.2.27 CPHY_Sync_IND

ASN.1 ASP Type Definition	
Type Name	CPHY_Sync_IND
PCO Type	CSAP
Comment	To indicate that physical channel synchronization (in FDD mode, sync with DPCCH) has been achieved.

Type Definition

```
SEQUENCE    {
               cellId                    INTEGER(0..15),
               routingInfo               RoutingInfo
}
```

7.3.2.2.28 CPHY_Out_of_Sync_IND

ASN.1 ASP Type Definition	
Type Name	CPHY_Out_of_Sync_IND
PCO Type	CSAP
Comment	To report that the physical channel synchronization (in FDD mode, sync with DPCCH) was lost as detected by the SS receiver.

Type Definition

```
SEQUENCE    {
               cellId                    INTEGER(0..15),
               routingInfo               RoutingInfo
}
```

7.3.2.2.29 CRLC_Resume_CNF

ASN.1 ASP Type Definition	
Type Name	CRLC_Resume_CNF
PCO Type	CSAP
Comment	To confirm the resume request
Type Definition	

```
SEQUENCE    {
               cellId                    INTEGER(0..15),
               routingInfo               RoutingInfo
}
```

7.3.2.2.30 CPHY_Frame_Number_REQ

ASN.1 ASP Type Definition	
Type Name	CPHY_Frame_Number_REQ
PCO Type	CSAP
Comment	To request the physical layer to return a connection frame number on which the next message can be sent at the specified PCO on the specified logical channel. The return frame number shall be far away from current frame number in order to leave some execution time for TTCN preparing next message.
Type Definition	

```
SEQUENCE    {
               cellId                    INTEGER(0..15),
               routingInfo               RoutingInfo
}
```

7.3.2.2.31 CPHY_Frame_Number_CNF

ASN.1 ASP Type Definition	
Type Name	CPHY_Frame_number_CNF
PCO Type	CSAP
Comment	To return the requested frame number
Type Definition	

```
SEQUENCE    {
                cellId                   INTEGER(0..15),
               routingInfo               RoutingInfo,
               frameNumber               INTEGER (0..255)
}
```

7.3.2.2.32 CMAC_SYSINFO_Config_REQ

ASN.1 ASP Type Definition	
Type Name	CMAC_SYSINFO_Config_REQ
PCO Type	CSAP
Comment	To request MAC layer to send the BCCH message on the specified configuration.
Type Definition	

```
SEQUENCE     {
               cellId                    INTEGER(0..15),
               routingInfo               RoutingInfo,
               ratType                   RatType,
               configMessage             CmacSysinfoConfigReq
}
```

ASN.1 Type Definition	
Type Name	CmacSysinfoConfigReq
Comment	

Type Definition

```
SEQUENCE    {
        SIM_ID                      INTEGER   {0..255},
        SG_REP                      INTEGER   (2..12),
        SG_POS                      INTEGER   {0..4096),
        BCCH_modificationTime       INTEGER   {0..4096}      OPTIONAL
}
```

7.3.2.2.33 CMAC_PAGING_Config_REQ

ASN.1 ASP Type Definition	
Type Name	CMAC_PAGING_Config_REQ
PCO Type	CSAP
Comment	To request MAC layer to send the Paging message on the specified configuration.

Type Definition

```
SEQUENCE     {
               cellId                    INTEGER(0..15),
               routingInfo               RoutingInfo,
               ratType                   RatType,
               configMessage             CmacPagingConfigReq
}
```

ASN.1 Type Definition	
Type Name	CmacPagingConfigReq
Comment	

Type Definition

```
SEQUENCE  {
             pI_BitMapInfo       CHOICE {
             e18                 BIT STRING (SIZE (18)),
             e36                 BIT STRING (SIZE (36)),
             e72                 BIT STRING (SIZE (72)),
             e144                BIT STRING (SIZE (144))},
             dRX_CycleLength     INTEGER  {2..12},
             iMSI                SEQUENCE (SIZE (6..15)) OF    Digit,
             t_pich_T_sccpch     BOOLEAN  --  T_pich>T_sccpch then FALSE
}
```

7.3.2.2.34 CRLC_SequenceNumber_REQ

ASN.1 ASP Type Definition	
Type Name	CRLC_SequenceNumber_REQ
PCO Type	CSAP
Comment	To request the RLC layer to return current counter sequence numbers

Type Definition

```
SEQUENCE    {
                cellId                    INTEGER(0..15),
                routingInfo               RoutingInfo
}
```

7.3.2.2.35 CRLC_SequenceNumber_CNF

ASN.1 ASP Type Definition	
Type Name	CRLC_Sequence_Number_CNF
PCO Type	CSAP
Comment	To return the requested counter sequence number
Type Definition	

```
SEQUENCE    {
                cellId                  INTEGER(0..15),
                routingInfo             RoutingInfo,
                count_C_MSB_UL          COUNT_C_MSB,
                count_C_MSB_DL          COUNT_C_MSB
}
```

7.3.2.2.36 CPHY_Cell_Config_REQ

ASN.1 ASP Type Definition	
Type Name	CPHY_Cell_Config_REQ
PCO Type	CSAP
Comment	To request to setup the cell parameter
Type Definition	

```
SEQUENCE    {
              cellId                    INTEGER(0..15),
              tcell                     INTEGER(0..38399),
              frequencyInfo             FrequencyInfo,
              primaryScramblingCode_SS  INTEGER(0..511),
              dLTxAttenuationLevel      INTEGER(0..30)
}
```

7.3.2.2.37 CPHY_Cell_Config_CNF

ASN.1 ASP Type Definition	
Type Name	CPHY_Cell_Config_CNF
PCO Type	CSAP
Comment	To confirm to setup the cell parameter
Type Definition	

```
SEQUENCE    {
            cellId                  INTEGER(0..15)
}
```

7.3.2.2.38 CMAC_SYSINFO_Config_CNF

ASN.1 ASP Type Definition	
Type Name	CMAC_SYSINFO_Config_CNF
PCO Type	CSAP
Comment	To confirm to setup the system information block
Type Definition	

```
SEQUENCE    {
            cellId                  INTEGER(0..15),
            routingInfo             RoutingInfo
}
```

7.3.2.2.39 CMAC_PAGING_Config_CNF

ASN.1 ASP Type Definition	
Type Name	CMAC_PAGING_Config_CNF
PCO Type	CSAP
Comment	To confirm to setup the paging message
Type Definition	

```
SEQUENCE    {
            cellId                      INTEGER(0..15),
            routingInfo                 RoutingInfo
}
```

7.3.2.2.40 CRLC_SecurityMode_Config_REQ

ASN.1 ASP Type Definition	
Type Name	CRLC_SecurityMode_Config_REQ
PCO Type	CSAP
Comment	To request to configure the RLC security mode
Type Definition	

```
SEQUENCE {
            cellId                      INTEGER(0..15),
            rlcSecurityInfo       SecurityInfo}
```

ASN.1 Type Definition	
Type Name	SecurityInfo
Comment	The integrityKey is not applicable to MAC
Type Definition	

```
SEQUENCE {
               cn-DomainIdentity                CN-DomainIdentity,
               cipheringKey                     BITSTRING(128),
               startValue                       START_VALUE,
               integrityKey                     BITSTRING(128)      OPTIONAL,
               gsmCiperingKey                   BITSTRING(64)       OPTIONAL
}
```

7.3.2.2.41 CMAC_SecurityMode_Config_REQ

ASN.1 ASP Type Definition	
Type Name	CMAC_SecurityMode_Config_REQ
PCO Type	CSAP
Comment	To request to configure the MAC security mode
Type Definition	

```
SEQUENCE {
            cellId                    INTEGER(0..15),
           macCipheringInfo      SecurityInfo
}
```

7.3.2.2.42 CRLC_Ciphering_Activate_REQ

ASN.1 ASP Type Definition	
Type Name	CRLC_Ciphering_Activate_REQ
PCO Type	CSAP
Comment	To request to start, restart or stop ciphering. RB-identity shall apply to routingInfo.Each call of the ASP includes one RLC SN in rb-DL-CiphActivationTimeInfo for the corresponding rb-identity.
Type Definition	

```
SEQUENCE {
            cellId                  INTEGER(0..15),
            routingInfo              RoutingInfo,
            ratType                  RatType,
            cipheringModeInfo        CipheringModeInfo
}
```

7.3.2.2.43 CMAC_Ciphering_Activate_REQ

ASN.1 ASP Type Definition	
Type Name	CMAC_Ciphering_Activate_REQ
PCO Type	CSAP
Comment	To request to start, restart or stop ciphering. The physicalChannelIdentity of DPCH applies to routingInfo.
Type Definition	

```
SEQUENCE {
            cellId                  INTEGER(0..15),
            routingInfo              RoutingInfo,
            ratType                  RatType,
            cipheringModeInfo        CipheringModeInfo
}
```

7.3.2.2.44 CRLC_SecurityMode_Config_CNF

ASN.1 ASP Type Definition	
Type Name	CRLC_SecurityMode_Config_CNF
PCO Type	CSAP
Comment	To confirm to configure the RLC security mode
Type Definition	

```
SEQUENCE {
            cellId                  INTEGER(0..15)
}
```

7.3.2.2.45 CMAC_SecurityMode_Config_CNF

ASN.1 ASP Type Definition	
Type Name	CMAC_SecurityMode_Config_CNF
PCO Type	CSAP
Comment	To confirm to configure the MAC security mode
Type Definition	

```
SEQUENCE {
            cellId                  INTEGER(0..15)
}
```

7.3.2.2.46 CRLC_Ciphering_Activate_CNF

ASN.1 ASP Type Definition	
Type Name	CRLC_Ciphering_Activate_CNF
PCO Type	CSAP
Comment	To confirm to activate or inactivate the ciphering
Type Definition	

SEQUENCE {
cellId INTEGER(0..15),
routingInfo RoutingInfo
}	

7.3.2.2.47 CMAC_Ciphering_Activate_CNF

ASN.1 ASP Type Definition	
Type Name	CMAC_Ciphering_Activate_CNF
PCO Type	CSAP
Comment	To confirm to activate or inactivate the ciphering
Type Definition	

SEQUENCE {
cellId INTEGER(0..15),
routingInfo RoutingInfo
}	

7.3.2.2.48 CRLC_Integrity_Activate_REQ

ASN.1 ASP Type Definition	
Type Name	CRLC_Integrity_Activate_REQ
PCO Type	CSAP
Comment	To request to start or to modifythe integrity protection. The ASP shall be called before send SECURITY MODE COMMAND. It activates the integrity on all SRBs in DL. Not to call the ASP if wishing to switch off the integrity in the test case.
Type Definition	

SEQUENCE {
cellId INTEGER(0..15),
integrityProtectionModeInfo IntegrityProtectionModeInfo
}	

7.3.2.2.49 CRLC_Integrity_Activate_CNF

ASN.1 ASP Type Definition	
Type Name	CRLC_integrity_Activate_CNF
PCO Type	CSAP
Comment	To confirm to activate or inactivate the integrity protection
Type Definition	

SEQUENCE {
cellId INTEGER(0..15)
}	

7.3.2.2.50 CMAC_SequenceNumber_REQ

ASN.1 ASP Type Definition	
Type Name	CMAC_SequenceNumber_REQ
PCO Type	CSAP
Comment	To request the MAC layer to return current counter sequence numbers. The physicalChannelIdentity of DPCH applies to routingInfo.

Type Definition

```
SEQUENCE    {
            cellId                      INTEGER(0..15),
            routingInfo                 RoutingInfo
}
```

7.3.2.2.51 CMAC_SequenceNumber_CNF

ASN.1 ASP Type Definition	
Type Name	CMAC_Sequence_Number_CNF
PCO Type	CSAP
Comment	To return the requested counter sequence number on MAC-d DCH. The physicalChannelIdentity of DPCH applies to routingInfo.

Type Definition

```
SEQUENCE    {
                  cellId                    INTEGER(0..15),
                  routingInfo               RoutingInfo,
                  count_C_MSB_UL            COUNT_C_MSB ,
                  count_C_MSB_DL            COUNT_C_MSB
}
```

7.3.2.2.52 CPHY_Cell_TxPower_Modify_REQ

ASN.1 ASP Type Definition	
Type Name	CPHY_Cell_TxPower_Modify_REQ
PCO Type	CSAP
Comment	To request to change the DL power

Type Definition

```
SEQUENCE    {
               cellId                   INTEGER(0..15),
               dLTxAttenuationLevel     INTEGER(0..30)
}
```

7.3.2.2.53 CPHY_Cell_TxPower_Modify_CNF

ASN.1 ASP Type Definition	
Type Name	CPHY_Cell_TxPower_Modify_CNF
PCO Type	CSAP
Comment	To confirm to change the DL power

Type Definition

```
SEQUENCE    {
               cellId                   INTEGER(0..15)
}
```

7.3.2.2.54 CRLC_Integrity_Failure_IND

ASN.1 ASP Type Definition	
Type Name	CRLC_Integrity_Failure_IND
PCO Type	CSAP
Comment	RLC emulator reports the occurrences of a failure in integrity protection, i.e. reception of an integrity-protected RLC AM/UM SDU containing a non-matching X-MAC value compared to the desired.
Type Definition	

```
SEQUENCE {
          cellId                    INTEGER(0..15),
          routingInfo               RoutingInfo,
          failureCause              ENUMERATED { codeNotMatched(0) }
          -- the enumerated types of failure cause field is ffs
}
```

7.3.3 Primitives for RLC and RB tests

The table below shows the primitives that are used for RLC and RB test, these primitives are defined in TTCN tabular form.

Table 15: Primitives for RLC and RB tests

Primitive	Parameters	Use
RLC_TR_TestDataReq	Cell identity Routing info Data (Meta type PDU)	The ASP is used to request the transmission of unstructured data using transparent mode in the downlink direction
RLC_TR_TestDataInd	Cell identity Routing info Data (Meta type PDU)	The ASP is used to indicate the reception of unstructured data using transparent mode in the uplink direction
RLC_UM_TestDataReq	Cell identity Routing info Data (Meta type PDU)	The ASP is used to request the transmission of unstructured data using unacknowledged mode in the downlink direction
RLC_UM_TestDataInd	Cell identity Routing info Data (Meta type PDU)	The ASP is used to indicate the reception of unstructured data using unacknowledged mode in the uplink direction
RLC_AM_TestDataReq	Cell identity Routing info Data (Meta type PDU)	The ASP is used to request the transmission of unstructured data using acknowledged mode in the downlink direction
RLC_AM_TestDataInd	Cell identity Routing info Data (Meta type PDU)	The ASP is used to indicate the reception of unstructured data using acknowledged mode in the uplink direction

The TTCN tabular format applies to the primitive definitions.

8 Design Consideration

8.1 Channel mapping

The figure 8 below shows the channel type mapping that is used for the configuration of the SS.

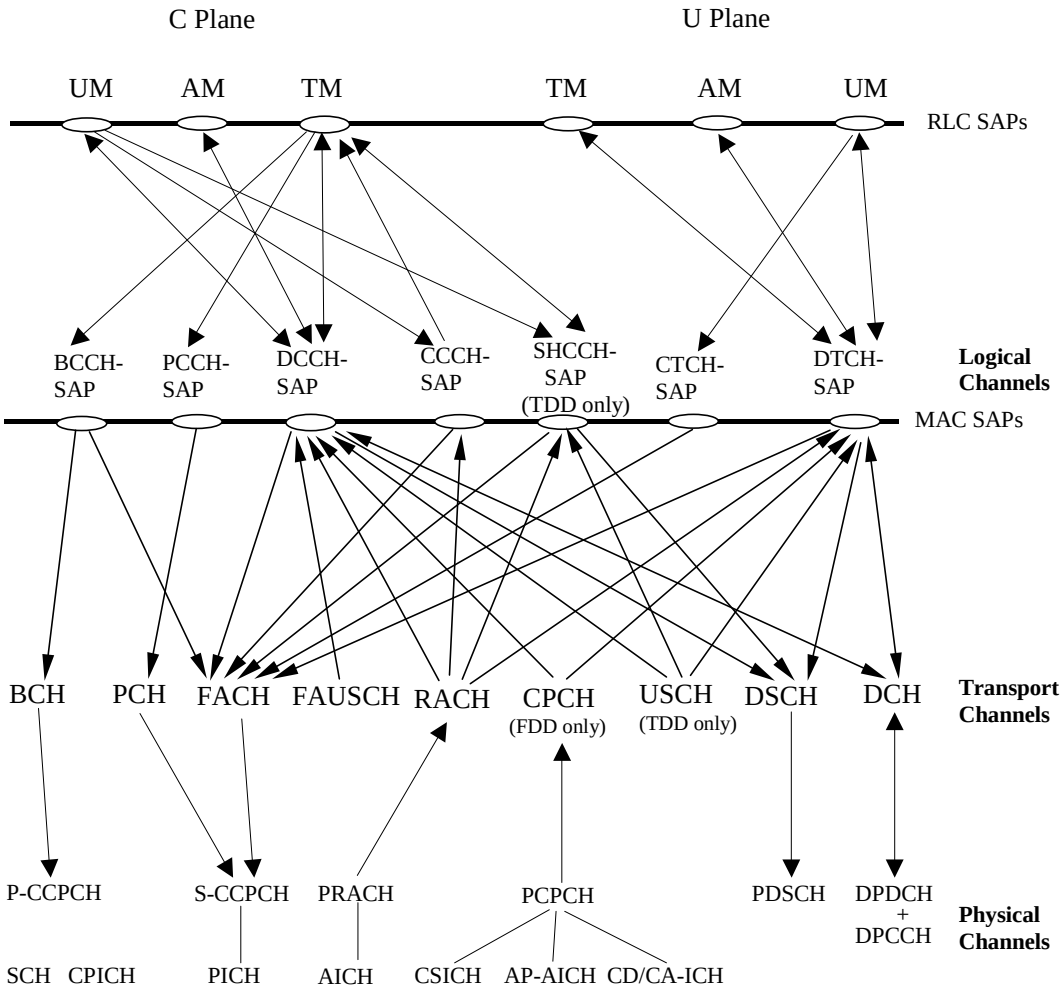

Figure 8: Channel mapping in SS

8.2 Channel and RB identity

The TTCN addresses the TTCN tester by using a channel identifier:

- Either Physical channel identifier (PhyCh id), or
- Transport channel identifier (TrCh id), or
- Radio bearer identifier (RB id).

The selected channel identifier identifies uniquely

- A channel within a cell,
- A total path of the address in the lower layers concerned.

Having taken out the cell id and PCO id (AM, UM and TM), a complete address, as RoutingInfo in the RRC ASP definition, should have at least five fields, CN domain id, RB id, LogCH id, TrCH id and PhyCH id. For simplified application of CHOICE of the routing information, a TTCN writer must carefully follow a number of rules assigning the channel identifiers.

General requirements:

A structured scheme of planning all channel identifiers assigned.

The scheme shall meet the requirements for all test cases in 34.123-1 including TDD channels.

The scheme can apply to all radio bearer configurations in 34.108, 6.10.

A clear multiplex mapping between a PhyCH id to TrCH ids and a TrCH id to LogCH ids, RB ids is needed.

Requirements on identification of RB in a test case:

Unique identification of the individual SRBs

Unique identification of the individual sub-flows of a RABs in CS and PS domain,

An assigned RB id can represent UL and DL.

Requirements on identification of TrCH in a test case:

Unique identification of the individual TrCH

Assign different identities for UL and DL of a same TrCH type

The order of the TrCH id assigned in a cell shall follow the TFCS definitions in the 34.108, 6.10. For example, TrCH ids are assigned in the ascending order for (RABsubflow#1, RABsubflow#2, RABsubflow#3, 64kRAB, DCCH).

Requirements on identification of PhyCH in a test case:

Unique identification of the individual PhyCH

Assign different identities for UL and DL of a same PhyCH type

Each S-CCPCH or PRACH has a unique identifier

For 2Mbps PS data (in case of demux of a TrCH), three DPCH or three DPDCH are needed for high-speed data. A single PhyCH id is assigned to a bundle of the three physical channels.

The table 16 shows which type of channel identity is chosen for the individual primitives.

Table 16: Primitieves and the associated channel identity type

Primitive name	Channel Idientity
CPHY_RL_Setup_REQ	PhysicalChannelIdentity
CHPY_TrCH_Config_REQ	PhysicalChannelIdentity
CMAC_Config_REQ	PhysicalChannelIdentity
CRLC_Config_REQ	RB_Identity

8.2.1 Physical Channels

Table 17: Physical channel identities

Type	Min. No.	Current Config.	Identities (value assigned)	Direction	Comment
P-CCPCH	1	1	tsc_P_CCPCH (4)	downlink	Primary Common Control Physical Channel. For Broadcasting System Information messages, using the Primary Scrambling Code for the Cell.
P-CPICH	1	1	tsc_P_CPICH (0)	downlink	Primary Common Pilot Channel using the Primary Scrambling Code for the Cell.
S-CPICH	1	FFS	tsc_S_CPICH (3)	downlink	Secondary Common Pilot Channel, used as the phase reference for some RF tests.
P-SCH	1	1	tsc_P_SCH (1)	downlink	Primary Synchronisation Channel
S-SCH	1	1	tsc_S_SCH (2)	downlink	Secondary Synchronisation Channel
S-CCPCH	2	1	tsc_S_CCPCH1 (5) tsc_S_CCPCH2 (10)	downlink	Secondary Common Control Physical Channel.
PICH	1	1	tsc_PICH1 (6)	downlink	To identify whether the UE should access the PCCH for Paging Messages.
AICH	1	1	tsc_AICH1 (7)	downlink	General Acquisition Indicator Channel, can be used for: - Aquisition Indicator Channel, for PRACH - Access Preamble Acquisition Indicator Channel (AP-ICH), for PCPCH - Collision-Detection/Channel-Assignment Indicator Channel (CD/CA-ICH), for PCPCH
DPCH	3	1	tsc_DL_DPCH1 (26) tsc_DL_DPCH2 (27)	downlink	Downlink Physical Data Channel. Layer 1 signalling is transmitted only on the first DPCH. This number is for the First Cell. Additional Cells may define a lower number which should be at least 1.
PDSCH	1	FFS		downlink	Physical Downlink Shared Channel.
DPDCH	1	1	tsc_UL_DPCH1 (20) tsc_UL_DPCH2 (21)	uplink	Uplink Dedicated Physical Channel. A single DPCCH associated with all the DPDCHs used for Layer 1 signalling.
PRACH	2	1	tsc_PRACH1 (8) tsc_PRACH2 (9)	uplink	Physical Random Access Channel.
PCPCH	1	FFS		uplink	Physical Common Packet Channel.
CSICH	1	FFS		downlink	CPCH Status Indicator Channel

The PhyCH values 20 to 25 are assigned to uplink DPCHs and the values 26 to 31 are assigned to downlink DPCHs,

8.2.2 Transport Channels

Table 18: Transport channel identities

Type	Min. No.	Current Config.	Identities (value assigned)	Direction	Comments
BCH	1	1	tsc_BCH1 (11)	downlink	
FACH	1	1	tsc_FACH1 (13) tsc_FACH2 (14)	downlink	
PCH	1	1	tsc_PCH1 (12)	downlink	
DCH	n	4	tsc_UL_DCH1 (1) tsc_UL_DCH2 (2) tsc_UL_DCH3 (3) tsc_UL_DCH4 (4) tsc_UL_DCH5 (5)	uplink	tsc_UL_DCH1 for RAB subflow#1, tsc_UL_DCH2 for RAB subflow#2, tsc_UL_DCH3 for RAB subflow#3, tsc_UL_DCH4 for future use, tsc_UL_DCH5 for SRB.
DCH	n	4	tsc_DL_DCH1 (6) tsc_DL_DCH2 (7) tsc_DL_DCH3 (8) tsc_DL_DCH4 (9) tsc_DL_DCH5 (10)	downlink	tsc_DL_DCH1 for RAB subflow#1, tsc_DL_DCH2 for RAB subflow#2, tsc_DL_DCH3 for RAB subflow#3, tsc_DL_DCH4 for future use, tsc_DL_DCH5 for SRB.
USCH	1	N/A	tsc_USCH1(20)	uplink	TDD only
DSCH	1	N/A	tsc_DSCH (19)	downlink	
RACH	2	1	tsc_RACH1 (15) tsc_RACH2 (16)	uplink	
CPCH	1	N/A	tsc_CPCH1(17)	uplink	
FAUSCH	N/A	N/A	tsc_FAUSCH1(18)	uplink	Not in Release 99

The TrCH values 20 – 29 are assigned to the TDD TrCH.

8.2.3 Logical Channels

The table below shows the logical channels identities

Table 19: Logical channel identities

Type	Min. No.	Current Config.	Identities (value assigned)	Direction	Comments
BCCH_BCH	1	1	tsc_BCCH1 (1)	downlink	
BCCH_FACH	1	1	tsc_BCCH5 (5)	downlink	
CCCH	1	1	tsc_CCCH6 (6)	downlink	
CCCH	1	1	tsc_CCCH5 (5)	uplink	
DCCH	4	4	tsc_DL_DCCH1 (1) tsc_DL_DCCH2 (2) tsc_DL_DCCH3 (3) tsc_DL_DCCH4 (4)	downlink	tsc_DL_DCCH1 for SRB1, tsc_DL_DCCH2 for SRB2, tsc_DL_DCCH3 for SRB3, tsc_DL_DCCH4 for SRB4
DCCH	4	4	tsc_UL_DCCH1 (1) tsc_UL_DCCH2 (2) tsc_UL_DCCH3 (3) tsc_UL_DCCH4 (4)	uplink	tsc_UL_DCCH1 for SRB1, tsc_UL_DCCH2 for SRB2, tsc_UL_DCCH3 for SRB3, tsc_UL_DCCH4 for SRB4
PCCH	1	1	tsc_PCCH1 (1)	downlink	
DTCH	n	4	tsc_UL_DTCH1 (7) tsc_UL_DTCH2 (8) tsc_UL_DTCH3 (9) tsc_UL_DTCH4 (10)	uplink	tsc_UL_DTCH1for RAB subflow#1, tsc_UL_DTCH2 for RAB subflow#2, tsc_UL_DTCH3 for RAB subflow#3' tsc_UL_DTCH4 for future use
DTCH	n	4	tsc_DL_DTCH1 (7) tsc_DL_DTCH2 (8) tsc_DL_DTCH3 (9) tsc_DL_DTCH4 (10)	downlink	tsc_DL_DTCH1for RAB subflow#1, tsc_DL_DTCH2 for RAB subflow#2, tsc_DL_DTCH3 for RAB subflow#3, tsc_DL_DTCH4 for future use
CTCH	1	1	tsc_CTCH (11)	downlink	

8.2.4 Radio bearers

Table 20: Radio bearer identities

Identities (value assigned)	Direction	Type	RLC mode	Service domain	Comments
tsc_RB_BCCH (-1)	downlink	SRB0	UM	NA	BCCH-BCH
tsc_RB0 (0)	uplink	SRB0	TM	CS or PS	The service domain for which the most recent security negotiation took place. CCCH
tsc_RB0 (0)	downlink	SRB0	UM	CS or PS	CCCH
tsc_RB1 (1)	uplink	SRB1	UM	CS or PS	DCCH
tsc_RB1 (1)	downlink	SRB1	UM	CS or PS	DCCH
tsc_RB2 (2)	uplink	SRB2	AM	CS or PS	DCCH
tsc_RB2 (2)	downlink	SRB2	AM	CS or PS	DCCH
tsc_RB3 (3)	uplink	SRB3	AM	CS or PS	DCCH
tsc_RB3 (3)	downlink	SRB3	AM	CS or PS	DCCH
tsc_RB4 (4)	uplink	SRB4	AM	CS or PS	DCCH
tsc_RB4 (4)	downlink	SRB4	AM	CS or PS	DCCH
tsc_RB5 (5)	uplink		TM		DCCH
tsc_RB5 (5)	downlink		TM		DCCH
tsc_RB_PCCH (7)	downlink		TM		PCCH PCH
tsc_RB9 (9)	downlink		TM		BCCH FACH
tsc_RB10 (10)	uplink	RAB#1	TM	CS	
tsc_RB10 (10)	downlink	RAB#1	TM	CS	
tsc_RB11 (11)	uplink	RAB#2	TM	CS	
tsc_RB11 (11)	downlink	RAB#2	TM	CS	
tsc_RB12 (12)	uplink	RAB#3	TM	CS	
tsc_RB12 (12)	downlink	RAB#3	TM	CS	
tsc_RB20 (20)	uplink	RAB#1	AM	PS	
tsc_RB20 (20)	downlink	RAB#1	AM	PS	
tsc_RB30 (30)	downlink		UM		CTCH FACH

The RB values 0-9 are used for the signalling bearers. The values 10-15 are assigned to the CS RAB sub-flows. The values 20-25 are assigned to the PS RAB sub-flows. The value 30 is assigned to the CBSMS/BMC service.

8.3 Channels configurations

8.3.1 Configuration of Cell_FACH

The configuration is based on 34.108, 6.10.2.4.3.2.1.2 for downlink and 34.108, 6.10.2.4.4.1.1.1 without RAB/DTCH for uplink. The configuration is applied to the RRC tests related in the states CELL_FACH, CELL_PCH and URA_PCH. They need a minimum radio configuration for testing.

Table 21: Uplink configuration of Cell_FACH

RB Identity	tsc_RB0 (0)	tsc_RB1 (1)	tsc_RB2 (2)	tsc_RB3 (3)	tsc_RB4 (4)
LogCh Type	CCCH	DCCH	DCCH	DCCH	DCCH
LogCh Identity	tsc_CCCH5 (5)	tsc_UL_DCCH1 (1)	tsc_UL_DCCH2 (2)	tsc_UL_DCCH3 (3)	tsc_UL_DCCH4 (4)
RLC mode	TM	UM	AM	AM	AM
TrCH Type	RACH				
TrCH identity	tsc_RACH1 (15)				
PhyCh Type	PRACH				
PhyCH identity	tsc_PRACH1 (8)				

Table 22: Downlink configuration of Cell_FACH

RB Identity	tsc_RB0 (0)	tsc_RB1 (1)	tsc_RB2 (2)	tsc_RB3 (3)	tsc_RB4 (4)	tsc_RB9 (9)	tsc_RB_PCCH (7)
LogCh Type	CCCH	DCCH	DCCH	DCCH	DCCH	BCCH	PCCH
LogCh Identity	tsc_CCCH6 (6)	tsc_DL_DCCH1 (1)	tsc_DL_DCCH2 (2)	tsc_DL_DCCH3 (3)	tsc_DL_DCCH4 (4)	tsc_BCCH5 (5)	tsc_PCCH1 (1)
RLC mode	UM	UM	AM	AM	AM	TM	TM
MAC priority	1	2	3	4	5	6	1
TrCH Type	FACH						PCH
TrCH identity	tsc_FACH2 (14)						tsc_PCH1 (13)
PhyCh Type	Secondary CCPCH						
PhyCH identity	tsc_S_CCPCH1 (5)						

8.3.2 Configuration of Cell_DCH_StandAloneSRB

The configuration is based on 34.108, 6.10.2.4.1.2. The RB0/UM-CCCH is referred to 34.108, 6.10.2.4.3.2.1.2 and RB0/TM-CCCH is referred to 34.108, 6.10.2.4.4.1.1.1. The configuration is applied to the RRC and NAS signalling tests in the DCH state without RAB.

Table 23: Uplink configuration of Cell_DCH_StandAloneSRB

RB Identity	tsc_RB1 (1)	tsc_RB2 (2)	tsc_RB3 (3)	tsc_RB4 (4)	tsc_RB0 (0)
LogCh Type	DCCH	DCCH	DCCH	DCCH	CCCH
LogCh Identity	tsc_UL_DCCH1 (1)	tsc_UL_DCCH2 (2)	tsc_UL_DCCH3 (3)	tsc_UL_DCCH4 (4)	tsc_CCCH5 (5)
RLC mode	UM	AM	AM	AM	TM
TrCH Type	DCH				RACH
TrCH identity	tsc_UL_DCH5 (5)				tsc_RACH1 (15)
PhyCh Type	DPDCH				PRACH
PhyCH identity	tsc_UL_DPCH1 (20)				tsc_PRACH1 (8)

Table 24: Downlink configuration of Cell_DCH_StandAloneSRB

RB Identity	tsc_RB1 (1)	tsc_RB2 (2)	tsc_RB3 (3)	tsc_RB4 (4)	tsc_RB0 (0)	tsc_RB_PCCH (7)
LogCh Type	DCCH	DCCH	DCCH	DCCH	CCCH	PCCH
LogCh Identity	tsc_DL_DCCH1 (1)	tsc_DL_DCCH2 (2)	tsc_DL_DCCH3 (3)	tsc_DL_DCCH4 (4)	tsc_CCCH6 (6)	tsc_PCCH1 (1)
RLC mode	UM	AM	AM	AM	UM	TM
MAC priority	1	2	3	4	1	1
TrCH Type	DCH				FACH	PCH
TrCH identity	tsc_DL_DCH5 (10)				tsc_FACH2 (14)	tsc_PCH1 (13)
PhyCh Type	DPCH				Secondary CCPCH	
PhyCH identity	tsc_DL_DPCH1 (20)				tsc_S_CCPCH1 (5)	

8.3.3 Configuration of Cell_DCH_Speech

The configuration is based on 34.108, 6.10.2.4.1.4. The RB0/UM-CCCH is referred to 34.108, 6.10.2.4.3.2.1.2 and RB0/TM-CCCH is referred to 34.108, 6.10.2.4.4.1.1.1. The configuration is applied to those RRC and NAS signalling tests in the DCH state where a CS voice service, such as narrowband speech, emergency speech call or TS 61 for speech, is established.

Table 25: Uplink configuration of Cell_DCH_Speech

RB Identity	tsc_RB10 (10)	tsc_RB11 (11)	tsc_RB12 (12)	Tsc_RB1 (1)	tsc_RB2 (2)	tsc_RB3 (3)	tsc_RB4 (4)	tsc_RB0 (0)
LogCh Type	DTCH	DTCH	DTCH	DCCH	DCCH	DCCH	DCCH	CCCH
LogCh Identity	tsc_UL_DTCH 1 (7)	tsc_UL_DTCH2 (8)	tsc_UL_DTCH3 (9)	tsc_UL_DCCH1 (1)	tsc_UL_DCCH2 (2)	tsc_UL_DCCH3 (3)	tsc_UL_DCCH4 (4)	tsc_CCCH5 (5)
RLC mode	TM	TM	TM	UM	AM	AM	AM	TM
TrCH Type	DCH	DCH	DCH	DCH				RACH
TrCH identity	tsc_UL_DCH1 (1)	tsc_UL_DCH2 (2)	tsc_UL_DCH3 (3)	tsc_UL_DCH5 (5)				tsc_RACH1 (15)
PhyCh Type	DPDCH							PRACH
PhyCH identity	tsc_UL_DPCH1 (20)							tsc_PRACH1 (8)

Table 26: Downplink configuration of Cell_DCH_Speech

RB Identity	tsc_RB10 (10)	tsc_RB11 (11)	tsc_RB12 (12)	Tsc_RB1 (1)	tsc_RB2 (2)	tsc_RB3 (3)	tsc_RB4 (4)	tsc_RB0 (0)	tsc_RB_PCCH (7)
LogCh Type	DTCH	DTCH	DTCH	DCCH	DCCH	DCCH	DCCH	CCCH	PCCH
LogCh Identity	tsc_DL_DTCH1 (7)	tsc_DL_DTCH2 (8)	tsc_DL_DTCH3 (9)	tsc_DL_DCCH1 (1)	tsc_DL_DCCH2 (2)	tsc_DL_DCCH3 (3)	tsc_DL_DCCH4 (4)	tsc_CCCH6 (6)	tsc_PCCH1 (1)
RLC mode	TM	TM	TM	UM	AM	AM	AM	UM	TM
MAC priority	1	1	1	1	2	3	4	1	1
TrCH Type	DCH	DCH	DCH	DCH				FACH	PCH
TrCH identity	tsc_DL_DCH1 (6)	tsc_DL_DCH2 (7)	tsc_DL_DCH3 (8)	tsc_DL_DCH5 (10)				tsc_FACH2 (14)	tsc_PCH1 (13)
PhyCh Type	DPCH							Secondary CCPCH	
PhyCH identity	tsc_DL_DPCH1 (20)							tsc_S_CCPCH1 (5)	

8.3.4 Configuration of Cell_DCH_64kCSRAB_SRB

The configuration is based on 34.108, 6.10.2.4.1.13 for the conversational unknown quality class. The RB0/UM-CCCH is referred to 34.108, 6.10.2.4.3.2.1.2 and RB0/TM-CCCH is referred to 34.108, 6.10.2.4.4.1.1.1. The configuration is applied to those RRC and NAS signalling tests in the DCH state where one of the following CS transparent data services is established:

- Multimedia call 28.8 k bit/s, 3.1kHz Audio,
- Multimedia call 32 k bit/s, UDI,
- Multimedia call 33.6 k bit/s, 3.1kHz Audio,
- Multimedia call 56 k bit/s, RDI,
- Multimedia call 64 k bit/s, UDI,
- Asynchronous 3.1kHz Audio 28.8 k bit/s,
- Synchronous 3.1kHz Audio 28.8 k bit/s,
- Synchronous V.110 UDI up to 56 k bit/s,
- BTM RDI 56 k bit/s,
- BTM UDI 64 bit/s.

Table 1627: Uplink configuration of Cell_DCH_64kCSRAB_SRB

RB Identity	tsc_RB10 (10)	tsc_RB1 (1)	tsc_RB2 (2)	tsc_RB3 (3)	tsc_RB4 (4)	tsc_RB0 (0)
LogCh Type	DTCH	DCCH	DCCH	DCCH	DCCH	CCCH
LogCh Identity	tsc_UL_DTCH1 (7)	tsc_UL_DCCH1 (1)	tsc_UL_DCCH2 (2)	tsc_UL_DCCH3 (3)	tsc_UL_DCCH4 (4)	tsc_CCCH5 (5)
RLC mode	TM	UM	AM	AM	AM	TM
TrCH Type	DCH	DCH				RACH
TrCH identity	tsc_UL_DCH1 (1)	tsc_UL_DCH5 (5)				tsc_RACH1 (15)
PhyCh Type	DPDCH					PRACH
PhyCH identity	tsc_UL_DPCH1 (20)					tsc_PRACH1 (8)

Table 28: Downlink configuration of Cell_DCH_64kCSRAB_SRB

RB Identity	tsc_RB10 (10)	tsc_RB1 (1)	tsc_RB2 (2)	tsc_RB3 (3)	tsc_RB4 (4)	tsc_RB0 (0)	tsc_RB_PCCH (7)
LogCh Type	DTCH	DCCH	DCCH	DCCH	DCCH	CCCH	PCCH
LogCh Identity	tsc_DL_DTCH1 (7)	tsc_DL_DCCH1 (1)	tsc_DL_DCCH2 (2)	tsc_DL_DCCH3 (3)	tsc_DL_DCCH4 (4)	tsc_CCCH6 (6)	tsc_PCCH1 (1)
RLC mode	TM	UM	AM	AM	AM	UM	TM
MAC priority	1	1	2	3	4	1	1
TrCH Type	DCH	DCH				FACH	PCH
TrCH identity	tsc_DL_DCH1 (6)	tsc_DL_DCH5 (10)				tsc_FACH2 (14)	tsc_PCH1 (13)
PhyCh Type	DPCH					Secondary CCPCH	
PhyCH identity	tsc_DL_DPCH1 (20)					tsc_S_CCPCH1 (5)	

8.3.5 Configuration of Cell_DCH_57_6kCSRAB_SRB

The configuration is based on 34.108, 6.10.2.4.1.17 for the streaming unknown quality class. The RB0/UM-CCCH is referred to 34.108, 6.10.2.4.3.2.1.2 and RB0/TM-CCCH is referred to 34.108, 6.10.2.4.4.1.1.1. The configuration is applied to those RRC and NAS signalling tests in the DCH state where one of the following CS non-transparent data services is established:

- Asynchronous 3.1kHz Audio up to 19.2 k bit/s,
- Asynchronous 3.1kHz Audio modem auto-bauding,
- Asynchronous V.110 UDI up to 38.4 k bit/s, except 28.8 k bit/s,
- Asynchronous V.120 up to 56 k bit/s,
- Asynchronous PIAFS up to 64 k bit/s,
- Asynchronous FTM up to 64 k bit/s,
- Synchronous 3.1kHz Audio up to 19.2 k bit/s,
- Synchronous V.110 UDI up to 56 k bit/s, except 28.8 k bit/s,
- Synchronous X.31 Flags Stuffing UDI up to 56 k bit/s,
- Synchronous V.120 up to 56 k bit/s,
- Synchronous BTM up to 64 k bit/s,
- TS61 FAX.

Table 29: Uplink configuration of Cell_DCH_57_6kCSRAB_SRB

RB Identity	tsc_RB10 (10)	tsc_RB1 (1)	tsc_RB2 (2)	tsc_RB3 (3)	tsc_RB4 (4)	tsc_RB0 (0)
LogCh Type	DTCH	DCCH	DCCH	DCCH	DCCH	CCCH
LogCh Identity	tsc_UL_DTCH1 (7)	tsc_UL_DCCH1 (1)	tsc_UL_DCCH2 (2)	tsc_UL_DCCH3 (3)	tsc_UL_DCCH4 (4)	tsc_CCCH5 (5)
RLC mode	TM	UM	AM	AM	AM	TM
TrCH Type	DCH	DCH				RACH
TrCH identity	tsc_UL_DCH1 (1)	tsc_UL_DCH5 (5)				tsc_RACH1 (15)
PhyCh Type	DPDCH					PRACH
PhyCH identity	tsc_UL_DPCH1 (20)					tsc_PRACH1 (8)

Table 30: Downlink configuration of Cell_DCH_57_6kCSRAB_SRB

RB Identity	tsc_RB10 (10)	tsc_RB1 (1)	tsc_RB2 (2)	tsc_RB3 (3)	tsc_RB4 (4)	tsc_RB0 (0)	tsc_RB_PCCH (7)
LogCh Type	DTCH	DCCH	DCCH	DCCH	DCCH	CCCH	PCCH
LogCh Identity	tsc_DL_DTCH1 (7)	tsc_DL_DCCH1 (1)	tsc_DL_DCCH2 (2)	tsc_DL_DCCH3 (3)	tsc_DL_DCCH4 (4)	tsc_CCCH6 (6)	tsc_PCCH1 (1)
RLC mode	TM	UM	AM	AM	AM	UM	TM
MAC priority	1	1	2	3	4	1	1
TrCH Type	DCH	DCH				FACH	PCH
TrCH identity	tsc_DL_DCH1 (6)	tsc_DL_DCH5 (10)				tsc_FACH2 (14)	tsc_PCH1 (13)
PhyCh Type	DPCH					Secondary CCPCH	
PhyCH identity	tsc_DL_DPCH1 (20)					tsc_S_CCPCH1 (5)	

8.3.6 Configuration of Cell_DCH_AM_UM_RAB_7Lis

The configuration is based on 34.123-1, table 7.2/1, 7.2/3A and 7.2/3B for the RLC AM and UM tests with 7 bit length indicators. The RB0/UM-CCCH is referred to 34.108, 6.10.2.4.3.2.1.2 and RB0/TM-CCCH is referred to 34.108, 6.10.2.4.4.1.1.1.

Table 31: Uplink configuration of Cell_DCH_AM_UM_RAB_7Lis

RB Identity	tsc_RB10 (10)	tsc_RB1 (1)	tsc_RB2 (2)	tsc_RB3 (3)	tsc_RB4 (4)	tsc_RB0 (0)
LogCh Type	DTCH	DCCH	DCCH	DCCH	DCCH	CCCH
LogCh Identity	tsc_UL_DTCH1 (7)	tsc_UL_DCCH1 (1)	tsc_UL_DCCH2 (2)	tsc_UL_DCCH3 (3)	tsc_UL_DCCH4 (4)	tsc_CCCH5 (5)
RLC mode	TM	UM	AM	AM	AM	TM
TrCH Type	DCH					RACH
TrCH identity	tsc_UL_DCH5 (5)					tsc_RACH1 (15)
PhyCh Type	DPDCH					PRACH
PhyCH identity	tsc_UL_DPCH1 (20)					tsc_PRACH1 (8)

Table 32: Downlink configuration of Cell_DCH_AM_UM_RAB_7Lis

RB Identity	tsc_RB10 (10)	tsc_RB1 (1)	tsc_RB2 (2)	tsc_RB3 (3)	tsc_RB4 (4)	tsc_RB0 (0)	tsc_RB_PCCH (7)
LogCh Type	DTCH	DCCH	DCCH	DCCH	DCCH	CCCH	PCCH
LogCh Identity	tsc_DL_DTCH1 (7)	tsc_DL_DCCH1 (1)	tsc_DL_DCCH2 (2)	tsc_DL_DCCH3 (3)	tsc_DL_DCCH4 (4)	tsc_CCCH6 (6)	tsc_PCCH1 (1)
RLC mode	TM	UM	AM	AM	AM	UM	TM
MAC priority	5	1	2	3	4	1	1
TrCH Type	DCH					FACH	PCH
TrCH identity	tsc_DL_DCH5 (10)					tsc_FACH2 (14)	tsc_PCH1 (13)
PhyCh Type	DPCH					Secondary CCPCH	
PhyCH identity	tsc_DL_DPCH1 (20)					tsc_S_CCPCH1 (5)	

8.3.7 Configuration of Cell_DCH_AMRAB_15Lis

The configuration is based on 34.123-1, table 7.2/4A and 7.2/4B for the RLC AM tests with 15 bit length indicators. The RB0/UM-CCCH is referred to 34.108, 6.10.2.4.3.2.1.2 and RB0/TM-CCCH is referred to 34.108, 6.10.2.4.4.1.1.1.

Table 33: Uplink configuration of Cell_DCH_AMRAB_15Lis

RB Identity	tsc_RB10 (10)	tsc_RB1 (1)	tsc_RB2 (2)	tsc_RB3 (3)	tsc_RB4 (4)	tsc_RB0 (0)
LogCh Type	DTCH	DCCH	DCCH	DCCH	DCCH	CCCH
LogCh Identity	tsc_UL_DTCH1 (7)	tsc_UL_DCCH1 (1)	tsc_UL_DCCH2 (2)	tsc_UL_DCCH3 (3)	tsc_UL_DCCH4 (4)	tsc_CCCH5 (5)
RLC mode	TM	UM	AM	AM	AM	TM
TrCH Type	DCH	DCH				RACH
TrCH identity	tsc_UL_DCH1 (1)	tsc_UL_DCH5 (5)				tsc_RACH1 (15)
PhyCh Type	DPDCH					PRACH
PhyCH identity	tsc_UL_DPCH1 (20)					tsc_PRACH1 (8)

Table 34: Downlink configuration of Cell_DCH_AMRAB_15Lis

RB Identity	tsc_RB10 (10)	tsc_RB1 (1)	tsc_RB2 (2)	tsc_RB3 (3)	tsc_RB4 (4)	tsc_RB0 (0)	tsc_RB_PCCH (7)
LogCh Type	DTCH	DCCH	DCCH	DCCH	DCCH	CCCH	PCCH
LogCh Identity	tsc_DL_DTCH1 (7)	tsc_DL_DCCH1 (1)	tsc_DL_DCCH2 (2)	tsc_DL_DCCH3 (3)	tsc_DL_DCCH4 (4)	tsc_CCCH6 (6)	tsc_PCCH1 (1)
RLC mode	TM	UM	AM	AM	AM	UM	TM
MAC priority	1	1	2	3	4	1	1
TrCH Type	DCH	DCH				FACH	PCH
TrCH identity	tsc_DL_DCH1 (6)	tsc_DL_DCH5 (10)				tsc_FACH2 (14)	tsc_PCH1 (13)
PhyCh Type	DPCH					Secondary CCPCH	
PhyCH identity	tsc_DL_DPCH1 (20)					tsc_S_CCPCH1 (5)	

8.3.8 Configuration of Cell_DCH_UMRAB_15Lis

The configuration is based on 34.123-1, table 7.2/2A and 7.2/2B for the RLC UM tests with 15 bit length indicators. The RB0/UM-CCCH is referred to 34.108, 6.10.2.4.3.2.1.2 and RB0/TM-CCCH is referred to 34.108, 6.10.2.4.4.1.1.1.

Table 35: Uplink configuration of Cell_DCH_UMRAB_15Lis

RB Identity	tsc_RB10 (10)	tsc_RB1 (1)	tsc_RB2 (2)	tsc_RB3 (3)	tsc_RB4 (4)	tsc_RB0 (0)
LogCh Type	DTCH	DCCH	DCCH	DCCH	DCCH	CCCH
LogCh Identity	tsc_UL_DTCH1 (7)	tsc_UL_DCCH1 (1)	tsc_UL_DCCH2 (2)	tsc_UL_DCCH3 (3)	tsc_UL_DCCH4 (4)	tsc_CCCH5 (5)
RLC mode	TM	UM	AM	AM	AM	TM
TrCH Type	DCH	DCH				RACH
TrCH identity	tsc_UL_DCH1 (1)	tsc_UL_DCH5 (5)				tsc_RACH1 (15)
PhyCh Type	DPDCH					PRACH
PhyCH identity	tsc_UL_DPCH1 (20)					tsc_PRACH1 (8)

Table 36: Downlink configuration of Cell_DCH_UMRAB_15Lis

RB Identity	tsc_RB10 (10)	tsc_RB1 (1)	tsc_RB2 (2)	tsc_RB3 (3)	tsc_RB4 (4)	tsc_RB0 (0)	tsc_RB_PCCH (7)
LogCh Type	DTCH	DCCH	DCCH	DCCH	DCCH	CCCH	PCCH
LogCh Identity	tsc_DL_DTCH1 (7)	tsc_DL_DCCH1 (1)	tsc_DL_DCCH2 (2)	tsc_DL_DCCH3 (3)	tsc_DL_DCCH4 (4)	tsc_CCCH6 (6)	tsc_PCCH1 (1)
RLC mode	TM	UM	AM	AM	AM	UM	TM
MAC priority	1	1	2	3	4	1	1
TrCH Type	DCH	DCH				FACH	PCH
TrCH identity	tsc_DL_DCH1 (6)	tsc_DL_DCH5 (10)				tsc_FACH2 (14)	tsc_PCH1 (13)
PhyCh Type	DPCH					Secondary CCPCH	
PhyCH identity	tsc_DL_DPCH1 (20)					tsc_S_CCPCH1 (5)	

8.3.9 Configuration of Cell_DCH_AMRAB_7_15Lis

The configuration is based on 34.123-1, 7.2.3.2.4 for the RLC AM tests involving selection of 7 or 15 bit length indicators. The RB0/UM-CCCH is referred to 34.108, 6.10.2.4.3.2.1.2 and RB0/TM-CCCH is referred to 34.108, 6.10.2.4.4.1.1.1.

Table 37: Uplink configuration of Cell_DCH_AMRAB_7_15Lis

RB Identity	tsc_RB10 (10)	tsc_RB1 (1)	tsc_RB2 (2)	tsc_RB3 (3)	tsc_RB4 (4)	tsc_RB0 (0)
LogCh Type	DTCH	DCCH	DCCH	DCCH	DCCH	CCCH
LogCh Identity	tsc_UL_DTCH1 (7)	tsc_UL_DCCH1 (1)	Tsc_UL_DCCH2 (2)	tsc_UL_DCCH3 (3)	tsc_UL_DCCH4 (4)	tsc_CCCH5 (5)
RLC mode	TM	UM	AM	AM	AM	TM
TrCH Type	DCH	DCH				RACH
TrCH identity	tsc_UL_DCH1 (1)	tsc_UL_DCH5 (5)				tsc_RACH1 (15)
PhyCh Type	DPDCH					PRACH
PhyCH identity	tsc_UL_DPCH1 (20)					tsc_PRACH1 (8)

Table 38: Downlink configuration of Cell_DCH_AMRAB_7_15Lis

RB Identity	tsc_RB10 (10)	tsc_RB1 (1)	tsc_RB2 (2)	tsc_RB3 (3)	tsc_RB4 (4)	tsc_RB0 (0)	tsc_RB_PCCH (7)
LogCh Type	DTCH	DCCH	DCCH	DCCH	DCCH	CCCH	PCCH
LogCh Identity	tsc_DL_DTCH1 (7)	tsc_DL_DCCH1 (1)	tsc_DL_DCCH2 (2)	tsc_DL_DCCH3 (3)	tsc_DL_DCCH4 (4)	tsc_CCCH6 (6)	tsc_PCCH1 (1)
RLC mode	TM	UM	AM	AM	AM	UM	TM
MAC priority	1	1	2	3	4	1	1
TrCH Type	DCH	DCH				FACH	PCH
TrCH identity	tsc_DL_DCH1 (6)	tsc_DL_DCH5 (10)				tsc_FACH2 (14)	tsc_PCH1 (13)
PhyCh Type	DPCH					Secondary CCPCH	
PhyCH identity	tsc_DL_DPCH1 (20)					tsc_S_CCPCH1 (5)	

8.3.10 Configuration of Cell_DCH_UMRAB_7_15Lis

The configuration is based on 34.123-1, 7.2.2.2.4 for the RLC UM tests involving selection of 7 or 15 bit length indicators. The RB0/UM-CCCH is referred to 34.108, 6.10.2.4.3.2.1.2 and RB0/TM-CCCH is referred to 34.108, 6.10.2.4.4.1.1.1.

Table 39: Uplink configuration of Cell_DCH_UMRAB_7_15Lis

RB Identity	tsc_RB10 (10)	tsc_RB1 (1)	tsc_RB2 (2)	tsc_RB3 (3)	tsc_RB4 (4)	tsc_RB0 (0)
LogCh Type	DTCH	DCCH	DCCH	DCCH	DCCH	CCCH
LogCh Identity	tsc_UL_DTCH1 (7)	tsc_UL_DCCH1 (1)	tsc_UL_DCCH2 (2)	tsc_UL_DCCH3 (3)	tsc_UL_DCCH4 (4)	tsc_CCCH5 (5)
RLC mode	TM	UM	AM	AM	AM	TM
TrCH Type	DCH	DCH				RACH
TrCH identity	tsc_UL_DCH1 (1)	tsc_UL_DCH5 (5)				tsc_RACH1 (15)
PhyCh Type	DPDCH					PRACH
PhyCH identity	tsc_UL_DPCH1 (20)					tsc_PRACH1 (8)

Table 40: Downlink configuration of Cell_DCH_UMRAB_7_15Lis

RB Identity	tsc_RB10 (10)	tsc_RB1 (1)	tsc_RB2 (2)	tsc_RB3 (3)	tsc_RB4 (4)	tsc_RB0 (0)	tsc_RB_PCCH (7)
LogCh Type	DTCH	DCCH	DCCH	DCCH	DCCH	CCCH	PCCH
LogCh Identity	tsc_DL_DTCH1 (7)	tsc_DL_DCCH1 (1)	tsc_DL_DCCH2 (2)	tsc_DL_DCCH3 (3)	tsc_DL_DCCH4 (4)	tsc_CCCH6 (6)	tsc_PCCH1 (1)
RLC mode	TM	UM	AM	AM	AM	UM	TM
MAC priority	1	1	2	3	4	1	1
TrCH Type	DCH	DCH				FACH	PCH
TrCH identity	tsc_DL_DCH1 (6)	tsc_DL_DCH5 (10)				tsc_FACH2 (14)	tsc_PCH1 (13)
PhyCh Type	DPCH					Secondary CCPCH	
PhyCH identity	tsc_DL_DPCH1 (20)					tsc_S_CCPCH1 (5)	

8.3.11 Configuration of Cell_FACH_BMC

The configuration is based on 34.108, 6.10.2.4.3.2.1.2 for downlink and 34.108, 6.10.2.4.4.1.1.1 without RAB/DTCH for uplink. A RB30/CTCH is configured. The configuration is applied to the BMC and CBSMS tests.

Table 41: Uplink configuration of Cell_FACH_BMC

RB Identity	tsc_RB0 (0)	tsc_RB1 (1)	tsc_RB2 (2)	Tsc_RB3 (3)	tsc_RB4 (4)
LogCh Type	CCCH	DCCH	DCCH	DCCH	DCCH
LogCh Identity	tsc_CCCH5 (5)	tsc_UL_DCCH1 (1)	tsc_UL_DCCH2 (2)	tsc_UL_DCCH3 (3)	tsc_UL_DCCH4 (4)
RLC mode	TM	UM	AM	AM	AM
TrCH Type	RACH				
TrCH identity	tsc_RACH1 (15)				
PhyCh Type	PRACH				
PhyCH identity	tsc_PRACH1 (8)				

Table 42: Downlink configuration of Cell_FACH_BMC

RB Identity	tsc_RB0 (0)	tsc_RB1 (1)	tsc_RB2 (2)	tsc_RB3 (3)	tsc_RB4 (4)	tsc_RB9 (9)	Tsc_RB30 (30)	tsc_RB_PCCH (7)
LogCh Type	CCCH	DCCH	DCCH	DCCH	DCCH	BCCH	CTCH	PCCH
LogCh Identity	tsc_CCCH6 (6)	tsc_DL_DCCH1 (1)	tsc_DL_DCCH2 (2)	tsc_DL_DCCH3 (3)	tsc_DL_DCCH4 (4)	tsc_BCCH5 (5)	Tsc_CTCH (11)	tsc_PCCH1 (1)
RLC mode	UM	UM	AM	AM	AM	TM	UM	TM
MAC priority	1	2	3	4	5	6	7	1
TrCH Type	FACH							PCH
TrCH identity	tsc_FACH2 (14)							tsc_PCH1 (13)
PhyCh Type	Secondary CCPCH							
PhyCH identity	tsc_S_CCPCH1 (5)							

8.3.12 Configuration of PS_Cell_FACH

The configuration is based on 34.108, 6.10.2.4.3.2.1.2 for downlink and 34.108, 6.10.2.4.4.1.1.1 for uplink. The configuration is applied to the RRC tests requiring a PS RAB in the state CELL_FACH.

Table 43: Uplink configuration of PS_Cell_FACH

RB Identity	tsc_RB20 (20)	tsc_RB0 (0)	tsc_RB1 (1)	tsc_RB2 (2)	tsc_RB3 (3)	tsc_RB4 (4)
LogCh Type	DTCH	CCCH	DCCH	DCCH	DCCH	DCCH
LogCh Identity	Tsc_UL_DTCH1 (7)	tsc_CCCH5 (5)	tsc_UL_DCCH1 (1)	tsc_UL_DCCH2 (2)	tsc_UL_DCCH3 (3)	tsc_UL_DCCH4 (4)
RLC mode	AM	TM	UM	AM	AM	AM
TrCH Type	RACH					
TrCH identity	tsc_RACH1 (15)					
PhyCh Type	PRACH					
PhyCH identity	tsc_PRACH1 (8)					

Table 44: Downlink configuration of PS_Cell_FACH

RB Identity	tsc_RB20 (20)	tsc_RB0 (0)	tsc_RB1 (1)	tsc_RB2 (2)	tsc_RB3 (3)	tsc_RB4 (4)	tsc_RB9 (9)	tsc_RB_PCH (7)
LogCh Type	DTCH	CCCH	DCCH	DCCH	DCCH	DCCH	BCCH	PCCH
LogCh Identity	tsc_DL_DTCH1 (6)	tsc_CCCH6 (6)	tsc_DL_DCCH1 (1)	tsc_DL_DCCH2 (2)	tsc_DL_DCCH3 (3)	tsc_DL_DCCH4 (4)	tsc_BCCH5 (5)	tsc_PCCH (1)
RLC mode	AM	UM	UM	AM	AM	AM	TM	TM
MAC priority	1	1	2	3	4	5	6	1
TrCH Type	FACH	FACH						PCH
TrCH identity	tsc_FACH1 (12)	tsc_FACH2 (14)						tsc_PCH (13)
PhyCh Type	Secondary CCPCH							
PhyCH identity	tsc_S_CCPCH1 (5)							

8.3.13 Configuration of PS Cell_DCH_64kPSRAB_SRB

The configuration is based on 34.108, 6.10.2.4.1.26. The RB0/UM-CCCH is referred to 34.108, 6.10.2.4.3.2.1.2 and RB0/TM-CCCH is referred to 34.108, 6.10.2.4.4.1.1.1. The configuration is applied to those RRC and NAS signalling tests in the DCH state where a PS RAB on DTCH is setup for the interactive or background service class.

Table 45: Uplink configuration of PS Cell_DCH_64kPSRAB_SRB

RB Identity	tsc_RB20 (20)	tsc_RB1 (1)	tsc_RB2 (2)	tsc_RB3 (3)	tsc_RB4 (4)	tsc_RB0 (0)
LogCh Type	DTCH	DCCH	DCCH	DCCH	DCCH	CCCH
LogCh Identity	tsc_UL_DTCH1 (7)	tsc_UL_DCCH1 (1)	tsc_UL_DCCH2 (2)	tsc_UL_DCCH3 (3)	tsc_UL_DCCH4 (4)	tsc_CCCH5 (5)
RLC mode	AM	UM	AM	AM	AM	TM
TrCH Type	DCH	DCH				RACH
TrCH identity	tsc_UL_DCH1 (1)	tsc_UL_DCH5 (5)				tsc_RACH1 (15)
PhyCh Type	DPDCH					PRACH
PhyCH identity	tsc_UL_DPCH1 (20)					tsc_PRACH1 (8)

Table 46: Downlink configuration of PS Cell_DCH_64kPSRAB_SRB

RB Identity	tsc_RB20 (20)	tsc_RB1 (1)	tsc_RB2 (2)	tsc_RB3 (3)	tsc_RB4 (4)	tsc_RB0 (0)	tsc_RB_PCCH (7)
LogCh Type	DTCH	DCCH	DCCH	DCCH	DCCH	CCCH	PCCH
LogCh Identity	tsc_DL_DTCH1 (7)	tsc_DL_DCCH1 (1)	tsc_DL_DCCH2 (2)	tsc_DL_DCCH3 (3)	tsc_DL_DCCH4 (4)	tsc_CCCH6 (6)	tsc_PCCH1 (1)
RLC mode	AM	UM	AM	AM	AM	UM	TM
MAC priority	1	1	2	3	4	1	1
TrCH Type	DCH	DCH				FACH	PCH
TrCH identity	tsc_DL_DCH1 (6)	tsc_DL_DCH5 (10)				tsc_FACH2 (14)	tsc_PCH1 (13)
PhyCh Type	DPCH					Secondary CCPCH	
PhyCH identity	tsc_DL_DPCH1 (20)					tsc_S_CCPCH1 (5)	

8.4 System information blocks scheduling

All SIBs specified in 34.108 are broadcast for all test cases in 34.123-3. The repeat period of broadcasting of a complete SIB configuration is 128 frames (1.28 s) for Europe and 64 frames (0.64s) for Japan.

Except MIB and SB1, they have the highest scheduling rates, SIB 2, 3, 4 and 7 have a higher scheduling rate.

The configurations of SIBs are regionally dependent, for example: only FDD-related SIBs needed in Japan, SIB16 in Europe, SIB 13 series and SIB 15 series in the North America. The schedule and SIB configurations are therefore different for a same test case. A PIXIT question is included in the ATSs to reflect the different regional requirements. Combination of one or several segments of variable length in the same SYSTEM INFORMATION message for testing is for further study.

According to the default SIB contents in 34.108, SB1, SIB1, SIB 11 and SIB12 have 2 segments. SIB 5 and SIB 6 have 3 segments. SIB 16 has 6 segments. MIB, SB1, SIB 2, SIB 3, SIB 4 and SIB 7 are not segmented, i.e. one segment for each.

Use CMAC_SYSINFO_CONFIG_REQ, CMAC_SYSINFO_CONFIG_CNF and RLC_TR_DATA_REQ as interface to SS for broadcasting.

Two TSOs are defined, one for PER encoding function, the other for segmentation function. The TSOs shall be implemented in the tester.

8.4.1 Grouping SIBs for testing

Mandatory in 34.108	**Used in Idle Mode**	MIB, SB1, (SB2), SIB1, SIB2, SIB3, SIB5, SIB7, SIB11
	Used in Connected Mode	SIB4, SIB6, SIB12
Mandatory for FDD CPCH		SIB8, SIB9
Mandatory for FDD DRAC		SIB10
Mandatory for TDD		SIB14, SIB17
Mandatory for LCS		SIB15, SIB15.1, SIB15.2, SIB15.3
Mandatory for ANSI-41 system		SIB13, SIB13.1, SIB13.2, SIB13.3, SIB13.4
Mandatory for InterSys HO		SIB16

8.4.2 SIB configurations

The ATS contains two SIB configurations, Configuration 1 for UTRAN/FDD SYSTEM and Configuration 2 for UTRAN/FDD and GERAN system. A PIXIT question related is raised for the selection of the SIB configuration for the test.

Configuration1)	MIB, SB1, SIB1, SIB2, SIB3, SIB4, SIB5, SIB6, SIB7, SIB11, SIB12
Configuration2	MIB, SB1, SIB1, SIB2, SIB3, SIB4, SIB5, SIB6, SIB7, SIB11, SIB12, SIB16

8.4.3 Test SIB schedule for UTRAN/FDD and GERAN system

Frame No.	0	2	4	6	8	10	12	14
REP-POS	0	1	2	3	4	5	6	7
Block Type	MIB	SB1	SB1	SIB3	MIB	SIB1	SIB1	SIB2

Frame No.	16	18	20	22	24	26	28	30
REP-POS	8	9	10	11	12	13	14	15
Block Type	MIB	SB1	SB1		MIB	SIB5	SIB5	SIB5

Frame No.	32	34	36	38	40	42	44	46
REP-POS	16	17	18	19	20	21	22	23
Block Type	MIB	SB1	SB1	SIB4	MIB	SIB6	SIB6	SIB6

Frame No.	48	50	52	54	56	58	60	62
REP-POS	24	25	26	27	28	29	30	31
Block Type	MIB	SB1	SB1	SIB7	MIB	SIB11	SIB11	

Frame No.	64	66	68	70	72	74	76	78
REP-POS	32	33	34	35	36	37	38	39
Block Type	MIB	SB1	SB1	SIB3	MIB	SIB16	SIB16	SIB16

Frame No.	80	82	84	86	88	90	92	94
REP-POS	40	41	42	43	44	45	46	47
Block Type	MIB	SB1	SB1	SIB16	MIB	SIB16	SIB16	

Frame No.	96	98	100	102	104	106	108	110
REP-POS	48	49	50	51	52	53	54	55
Block Type	MIB	SB1	SB1	SIB4	MIB	SIB12	SIB12	

Frame No.	112	114	116	118	120	122	124	126
REP-POS	56	57	58	59	60	61	62	63
Block Type	MIB	SB1	SB1	SIB7	MIB			

SIB-repeat period (in frame)

Block Type	MIB	SB1	SIB1	SIB2	SIB3	SIB4	SIB5	SIB6	SIB7	SIB11	SIB12	SIB16
SIB Rep	8	16	128	128	64	64	128	128	64	128	128	128
Max. No of seg.	1	2	2	1	1	1	3	3	1	2	2	6

8.4.4 Test SIB schedule for UTRAN FDD system

Frame No.	0	2	4	6	8	10	12	14
REP-POS	0	1	2	3	4	5	6	7
Block Type	MIB	SB1	SB1	SIB3	MIB	SIB1	SIB1	SIB2

Frame No.	16	18	20	22	24	26	28	30
REP-POS	8	9	10	11	12	13	14	15
Block Type	MIB	SB1	SB1	SIB7	MIB	SIB5	SIB5	SIB5

Frame No.	32	34	36	38	40	42	44	46
REP-POS	16	17	18	19	20	21	22	23
Block Type	MIB	SB1	SB1	SIB4	MIB	SIB6	SIB6	SIB6

Frame No.	48	50	52	54	56	58	60	62
REP-POS	24	25	26	27	28	29	30	31
Block Type	MIB	SB1	SB1	SIB11	MIB	SIB11	SIB12	SIB12

SIB-repeat period (in frame)

Block Type	MIB	SB1	SIB1	SIB2	SIB3	SIB4	SIB5	SIB6	SIB7	SIB11	SIB12
SIB Rep	8	16	64	64	64	64	64	64	64	64	64
Max. No of seg	1	2	2	1	1	1	3	3	1	2	2

8.4.5 Handling the transmission of SIB

According to the SIB repeat periods, SIBs need to be transmitted on a very regular basis during the operation of a test case. This transmission usually has no direct bearing on the operation of the test case, although the carried information ensures the correct configuration and operation of the UE during the test case.

To send this information repeatedly directly from each test case would make the test cases very complex to implement, difficult to understand and place real-time requirements upon them that are beyond the capabilities of most TTCN driven test engines.

Management of scheduling of System Information messages is performed by the System simulator. The SIB contents, usually determined in part by the individual tests, come from the TTCN test cases.

8.4.5.1 Delivery of System Information content

The content of the System Information messages is delivered as a fully encoded bit string to the TM-RLC SAP from the message content defined in the TTCN test case.

The IE ‘SFNprime’ in the SI messages is set to 0 by the TTCN, and the correct value of ‘SFNprime’ shall be inserted by the System Simulator prior to transmission of a SI message.

SI messages are ASN.1 packed encoded through a TTCN TSO and segmented another TTCN TSO into SIBs in the TTCN and sent only once to the TM-RLC SAP. Repetition of the SIB is the responsibility of the System Simulator lower layers.

SIBs are considered to be cached. That is, sending a SIB to the TM-RLC SAP will cause a previously sent copy of the SIB to be lost, and all future transmissions of the SIB will be the most recently sent version. This allows for the updating of System Information during the operation of a test case.

8.4.5.2 Scheduling of System Information Blocks

The schedule for the transmission of SIBs is provided by the TTCN test case. It is sent using the CMAC_SYSINFO_CONFIG_REQ primitive sent to the CMAC SAP (CMAC_PCO).

Each CMAC_SYSINFO_CONFIG_REQ primitive carries scheduling information for the next SIB sent from the TTCN. Each primitive is followed by an associated SIB. Sending two CMAC_SYSINFO_CONFIG_REQ primitives in succession may cause an unspecified result.

8.4.5.3 Example of usage

The following sequence shows how some SIBs are sent to the System Simulator lower layers, together with scheduling information:

CMAC_PCO: CMAC_SYSINFO_CONFIG_REQ (8, 0)
TM_PCO: MIB
CMAC_PCO: CMAC_SYSINFO_CONFIG_REQ (16, 1)
TM_PCO: SB1 (segment 1 of 2)
CMAC_PCO: CMAC_SYSINFO_CONFIG_REQ (16, 2)
TM_PCO: SB1 (segment 2 of 2)
CMAC_PCO: CMAC_SYSINFO_CONFIG_REQ (64, 3)
TM_PCO: SIB3 (segment 1 of 1)
CMAC_PCO: CMAC_SYSINFO_CONFIG_REQ (8, 4)
TM_PCO: MIB
CMAC_PCO: CMAC_SYSINFO_CONFIG_REQ (128, 5)
TM_PCO: SIB1 (segment 1 of 2)
…

8.5 Security in testing

The security functions in the SS side are implemented in RLC and MAC layers. The NAS and RRC TTCN relevant test steps communicate with the L2 emulator through PIXIT, TSOs and ASPs.

8.5.1 Authentication

A GMM or MM authentication test step makes use of a number of TSOs to generate an authentication vector:

AV := {RAND, XRES, CK, IK, AUTN}

8.5.2 Ciphering

tbd

8.5.3 Integrity

tbd

8.5.4 Counter check

tbd

8.5.5 Test USIM

tbd

8.6 Downlink power control in SS

tbd

8.7 Test suite operation definitions

Table 47: TSO definitions

TSO Name	Description

o_AuthRspChk	**Type of the result:** BOOLEAN
	Parameters:
	p_AuthRsp : AuthRsp
	p_AuthRspExt : AuthRspExt
	p_Ki : BITSTRING
	p_RAND : BITSTRING
	p_Ext : BOOLEAN
	
	Description
	Checks the input parameter p_AuthRsp and p_AuthRspExt, both received in an Authentication Response, according to the authentication algorithm defined in the following procedure.
	The extension, p_AuthRspExt, is optional. Its presence is indicated by p_Ext.
	Returns TRUE if the Authentication Response contained in parameters p_AuthRsp and eventually p_AuthRspExt is correct, FALSE otherwise.
	The value of tcv_Auth_n indicates whether the AuthRspExt has been provided by the UE or not (n=31, or 31 < n < 128). See 3GPP TS 34.108 cl. 8.1.2.
	If not the parameter p_AuthRspExt is not to be used.
	
	Algorithm (without the knowledge of tcv_Auth_n):
	==========
	if NOT p_Ext EvaluateAuthRsp else EvaluateAuthRspAndAuthRspExt
	EvaluateAuthRsp:
	==========
	resultbitstring = o_BitstringXOR(XRES, AuthRsp)
	if resultbitstring is all 0s then there is a match.
	
	EvaluateAuthRspAndAuthRspExt:
	==========
	XREShigh = o_BitstringXtract(XRES, 32, 32, 0)
	/* XRES divides into 2 parts: the higher part of 32 bits related to AuthRsp and the lower part related to AuthRspExt */
	/* SourceLength of 32 is only to ensure usage of the procedure */
	resultbitstring = o_BitstringXOR(XREShigh, AuthRsp)
	if resultbitstring is all 0s then there is a match for the first 32 bits:EvaluateAuthRspExt
	else Authentication failed.
	
	EvaluateAuthRspExt:
	==========
	/* As AuthRespExt may not be octet aligned the last octet indicated in AuthRspExt is not used for checking */
	if (AuthRspExt.iel = 1)
	then Authentication passed
	/* there was only 1 possibly incomplete octet which is not used */
	else
	{
	AuthRspExthigh = o_BitstringXtract(AuthRspExt.authRsp, ((AuthRspExt.iel -1)* 8), (AuthRspExt.iel -1)* 8, 0)
	/* extract (AuthRspExt.iel -1)* 8 bits starting from bit 0 */
	XRESlow = o_BitstringXtract(XRES, ((AuthRspExt.iel -1)* 8 + 32), (AuthRspExt.iel -1)* 8, 32)
	/* extract (AuthRspExt.iel -1)* 8 bits starting from bit 32 */
	resultbitstring = o_BitstringXOR(XRESlow, AuthRspExthigh, (AuthRspExt.iel -1)* 8)
	if resultbitstring is all 0s then there is a match for the bits following the first 32 bits else Authentication failed

o_AuthSQN_Upd	**Type of the result:** BOOLEAN Increments BITSTRING tsv_SQN so that it becomes available for use by the next authentication. Returns TRUE if the procedure succeds, FALSE otherwise. Note : the incremenation algorithm is to be specified. Not yet known. ???
o_BitstringChange	**Type of the result:** BITSTRING **Parameters:**

	P_Str: BITSTRING p_Len: INTEGER p_Offset: INTEGER **Description** Performs the manipulation of a bitstring by toggling the bit identified by p_Offset. The length of the string to be manipulated is specified in p_Len. This is only provided to help ensure that the p_Offset is less than p_Len. Returns a resulting bitstring of length p_Len. Examples: o_BitstringChange('010101'B, 6, 5) produces '010100'B. o_BitstringChange('010101'B, 6, 0) produces '110101'B.
o_BitstringConcat	**Type of the result:** BITSTRING **Parameters:** P_Str1: BITSTRING p_Str2: BITSTRING p_Len1: INTEGER p_Len2: INTEGER **Description** Performs the concatenation of 2 bitstrings of possibly different lengths. The bit significance is from left to right, ie the MSB is at the lefthand side. Returns a resulting bitstring p_Str1 || p_Str2 of length p_ Len1 + p_Len. Example: o_BitstringConcat('010101'B,'11'B) produces '01010111'B of length 6 + 2 = 8.
o_BitstringXtract	**Type of the result:** BITSTRING **Parameters:** P_Str: BITSTRING p_SrcLen: INTEGER p_TargetLen: INTEGER p_Offset: INTEGER **Description** Performs the wrap around extract of a bitstring. The length of the string from which extraction is to be made is specified in p_SrcLen. The length of the bitstring to be extracted is indicated as p_TargetLen, the offset in the original string is indicated in p_Offset. The bit significance is from left to right, ie the MSB is at the lefthand side. Returns a resulting bitstring of length p_TargetLen. Examples: o_BitstringXtract('010101'B, 6, 2, 1) produces '10'B. o_BitstringXtract('010101'B, 6, 4, 3) produces '1010'B, wrapping around. o_BitstringXtract('000111'B, 6, 4, 3) produces '1110'B, wrapping around.
o_BitstringXOR	**Type of the result:** BITSTRING **Parameters:** P_Str1: BITSTRING p_Str2: BITSTRING p_Len: INTEGER **Description** Performs an XOR operation using 2 bitstrings of the same length (p_Len). Returns a resulting Bitstring of length p_Len. Example: o_BitstringXOR('0011'B, '0101'B, 4) produces '0110'B
o_ConcatStrg	**Type of the result:** IA5String **Parameters:** P_String1: IA5String p_String2: IA5String **Description** o_ConcatString concatenates 'p_String1' and 'p_String2' and returns the resulting string. For example: o_ConcatString ("AT+CBST=0" , ",0") = "AT+CBST=0,0"
o_ConvertIMSI	**Type of the result:** IMSI_GSM_MAP **Parameters:** P_Imsi : HEXSTRING The input parameter `p_Imsi` is a BCD string (subset of HEXSTRING), the result is of type IMSI_GSM_MAP.

o_FirstDigit	**Type of the result:** B4 **Parameters:** p_BCDdigits : HEXSTRING **Description** The input parameter p_BCDdigits shall be a BCD string (subset of HEXSTRING), the resut is a BITSTRING[4] of a binary representation of one BCD digit. The function of the o_FirstDigit is to return the first (most significant) digit of the input parameter ` p_BCDdigits `. for example: o_FirstDigit('12345') = '0001'B, o_FirstDigit('012345678') = '0000'B.
o_GetPI	**Type of the result:** B18 **Parameters:** p_Imsi : HEXSTRING p_DrxCycle: INTEGER p_Pbp: INTEGER p_Np: INTEGER **Description** The operation is used to calculate the PI (Page Indicator) from the given input parameters. The PI is calculated as following: PI = drx_index mod np The drx_index is calculated as described hereafter: drx_index = (p_Imsi / (p_DrxCycle / p_Pbp))
o_HexToIA5	**Type of the result:** IA5String **Parameters:** p_String: HEXSTRING **Description** o_HEX_TO_IA5 converts hexadecimal string 'p_String' to an IA5 String For example: o_HEX_TO_IA5 ('15A'H) = "15A"

o_IntToOct	**Type of the result:**OCTETSTRING
	Parameters:
	p_N : INTEGER
	p_L: INTEGER
	
	Description
	o_IntToOct converts the INTEGER `p_N` into OCTETSTRING with length = `p_L`.
	for example:
	o_IntToOct(14,1) = '0E'O;
	o_IntToOct(18,1) = '12'O;
	o_IntToOct(18,2) = '0012'O.

o_OeBit	**Type of the result:**BITSTRING **Parameters:** p_BCDdigits: HEXSTRING **Description** The input parameter ` p_BCDdigits ` is a BCD string (subset of HEXSTRING), the result is BITSTRING[1]. The function of the o_OeBit is as the follows: 1. it returns '1'B, if the length of the ` p_BCDdigits ` is odd, 2. it returns '0'B, if the length of the ` p_BCDdigits ` is even. for example: o_OeBit('12583') = '1'B, o_OeBit('87259957') ='0'B.
o_OtherDigits	**Type of the result:**OCTETSTRING **Parameters:** p_BCDdigits : HEXSTRING The input parameter ` p_BCDdigits ` is a BCD string (subset of HEXSTRING), the result is an even string of BCD digits, with eventually a filler 'F'H used. */ The function of the o_OtherDigits is as the follows: 1. If the number of the ` p_BCDdigits ` is odd, the operation removes the most significant digit, and then reverses the order of each pair of digits; 2. If the number of the ` p_BCDdigits ` is even, first the operation suffixes the `bcddigits`

	with 'F'H, then removes the most significant digit, and then reverses the order of each pair of digits. for example: o_OtherDigi('12345') = '3254', o_OtherDigi('12345678') ='325476F8'. See o_FirstDigit for the handling of the first digit.
o_CheckStringStartWith	**Type of the result:**BOOLEAN **Parameters:** p_SourceString: IA5String p_StartString : IA5String **Description** o_CheckStringStartWith returns TRUE if the p_sourceString start with the p_StartString. Otherwise it returns FALSE. For example: o_CheckStringStartWith ("+CLCC:1,0,0,2,0;", "+CLCC:1,0,0")=TRUE */
o_IA5ToOct	**Type of the result:**OCTETSTRING **Parameters:** p_String : IA5String **Description** o_IA5_TO_OCT convert the string 'p_String' from IA5String type to OCTETSTRING. o_IA5_TO_OCT shall use '1111'B as an end mark, in the case of an odd number ofnumber digits. For example: o_IA5_TO_OCT ("15A") = '15AF'O
o_GetN_OctetsFromPRBS	**Type of the result:**OCTETSTRING **Parameters:** p_Start, p_N: INTEGER **Description** This operation returns N octets from a repeated pseudo random bit sequence, starting with octet position p_Start. The PRBS is the 2047 bit pseudo random test pattern defined in ITU-T Recommendation O.153 for measurements at 64 kbit/s and N x 64 kbit/s o_GetN_OctetsFromPRBS(p_Start, p_N) generates an OCTETSTRING containing p_N octets starting from octet number p_Start in the PRBS. Requirements p_Start >= 0 p_N >= 1 Definition Define the 2047 bit PRBS sequence b(i) as an m-sequence produced by using the following primitive (over GF(2)) generator polynomial of degree 11: X^11 + X^9 + 1 This sequence is defined recursively as: b(i) = 1 , i = 0,1,...,10 b(i) = b(i - 2) + b(i - 11) modulo 2 , i = 11,16,...,2046 The OCTETSTRING, o(j) generated by the present TSO is produced by extracting p_N octets from the repeated sequence b(i) as follows: o(j,k) = b(((n_Start + j) * 8 + k) modulo 2047) where: j = 0,1,..,p_N - 1 k = 0,1,..7 o(j,k) is the kth bit of the jth octet in o(j), o(j,0) is the MSB of the jth octet in o(j), o(j,7) is the LSB of the jth octet in o(j), Example results: o_GetN_OctetsFromPRBS(0, 25) and o_GetN_OctetsFromPRBS(2047, 25) both return: 'FFE665A5C5CA3452085408ABEECE4B0B813FD337873F2CD1E2'O o_GetN_OctetsFromPRBS(255, 25) and o_GetN_OctetsFromPRBS(255 + 2047, 25) both return '01FFCCCB4B8B9468A410A81157DD9C9617027FA66F0E7E59A3'O
o_SIB_PER_Encoding	**Type of the result:**BITSTRING **Parameters:** p_SIB : SIB **Description** It returns the unaligned PER encoding (BIT STRING) of the input system information

	block p_SIB. Example: for the following SIBType1 value: SysInfoType1 ::= { cn-CommonGSM-MAP-NAS-SysInfo '32F4100001'H, cn-DomainSysInfoList { { cn-DomainIdentity ps-domain, cn-Type gsm-MAP : '0000'H, cn-DRX-CycleLengthCoeff 7}, {cn-DomainIdentity cs-domain, cn-Type gsm-MAP : '0001'H, cn-DRX-CycleLengthCoeff 7}}, ue-IdleTimersAndConstants { t-300 ms400, n-300 7, t-312 10, n-312 s200}, ue-DCHTimersAndConstants { t-304 ms100, n-304 7, t-308 ms40, t-309 8, t-313 15, n-313 s200, t-314 s20, t-315 s1800, n-315 s1000}, nonCriticalExtensions { } } The operation returns BITSTRING: "0100001100101111010000001000000000000000000010101000100000000000000010000001000000000000000100100101111010011010100011111111111111111111111111111F"
o_SIBSegmentation	**Type of the result:**SegmentsOfSysInfoBlock **Parameters:** p_SIBBitString : BITSTRING **Description** Function of o_SIBSegmentation is as following: 1. The input operand p_SIBBitString is segmented from left to right into segments, each segment except the last one is 222 bits . The last segment may be 222 bits or shorter. If the length of last segment is great than 214 bits pad it to 222 bits with padding bits set to '0'. 2. The number of segments is assigned to segCount field of the result. 3. The first segment is assigned to seg1 field of the result, the second segment is assigned to the seg2 field of the result, the third segment is assigned to the seg3 field of the result, and so on till the last segment.
o_HexToDigitsMNC	**Type of the result:**MNC **Parameters:** p_BCDdigits : HEXSTRING **Description** The function of this operation is: 1. the least significant HEX is removed if it is 'F' and the operation returns SEQUENCE (SIZE(2)) OF Digit. 2. the operation returns SEQUENCE (SIZE(3)) OF Digit if all 3 HEX digits in p_BCDdigits are BCD Digit. for example: o_HexToDigitsMNC('123'H) = {1, 2, 3} o_HexToDigitsMNC('13F'H) = {1, 3}.
o_HexToDigitsMCC	**Type of the result:**MCC **Parameters:** p_BCDdigits : HEXSTRING **Description** The input parameter p_BCDdigits shall be a BCD string (subset of HEXSTRING), the result is a SEQUENCE (SIZE(3)) OF digit (MCC). NOTE: The length of p_BCDdigits shall be 3. User shall take the responsibility of fulfilling

	this requirement. for example: o_HexToDigitsMCC('111'H) = {1, 1, 1} o_HexToDigitsMCC('123'H) = {1, 2, 3}.
o_ConvtAndConcatStr	**Type of the result:**OCTETSTRING **Parameters:** p_MCC, p_MNC : HEXSTRING; p_LAC : OCTETSTRING **Description** functions of o_ConvtAndConcatStr are as following: 1. The least significant HEX of p_MNC is removed from p_MNC and inserted into p_MCC in the position left to the third HEX to form a new p_MCC of 4 HEXs, then swap the first HEX (left most, most siginificant Hex) with the second HEX of the new p_MCC. 2. Swap the first Hex with the second HEX of the remaining part of p_MNC and append it to the new p_MCC formed in step1 above. 3. Append p_LAC to the result of step 2, this is the final result.
o_OctetstringConcat	**Type of the result:**OCTETSTRING **Parameters:** p_Str1, p_Str2: OCTETSTRING **Description** o_OctetstringConcat Performs the concatenation of 2 octetstrings of possibly different lengths. The octet significance is from left to right, i.e. the MSB is at the lefthand side. Returns a resulting octetstring p_Str1 || p_Str2. Example: o_OctetstringConcat('135'O, '9A38'O) = '1359A38'O.
o_ConvertTMSI	**Type of the result:**TMSI_GSM_MAP **Parameters:** p_Tmsi : OCTETSTRING **Description** The input parameter `p_Tmsi` is an OCTETSTRING; the result is of type TMSI_GSM_MAP.

8.8 AT commands

The table below shows a list of AT commands. By using these commands the ATSs communicate with the SS for an automatic execution. The column ‘ATS’ indicates in which ATS the command is used.

Table 48: AT commands used in 3GPP ATSs

Command	Reference	ATS
A	TS 27.007 [23]	RRC, NAS, SMS
+CMOD	TS 27.007 [23]	RRC, NAS, SMS
+CBST	TS 27.007 [23]	RRC, NAS, SMS
D	TS 27.007 [23]	RRC, NAS, SMS
H	TS 27.007 [23]	NAS
+CLCC	TS 27.007 [23]	NAS
+VTS	TS 27.007 [23]	NAS
+CGMF	TS 27.005 [22]	SMS
+CGMD	TS 27.005 [22]	SMS
+CGMR	TS 27.005 [22]	SMS
+CMGW	TS 27.005 [22]	SMS
+CMSS	TS 27.005 [22]	SMS
+CNMI	TS 27.005 [22]	SMS
+CPMS	TS 27.005 [22]	SMS
+CSMS	TS 27.005 [22]	SMS
+CSMP	TS 27.005 [22]	SMS
+CSCS	TS 27.005 [22]	SMS
+CSCA	TS 27.005 [22]	SMS
+CGDCONT	TS 27.007 [23]	NAS
+CGEQREQ	TS 27.007 [23]	NAS
+CGEREQMIN	TS 27.007 [23]	NAS
+CGDSCONT	TS 27.007 [23]	NAS

Annex A (normative): Abstract Test Suites (ATS)

This annex contains the approved ATSs.

The ATSs have been produced using the Tree and Tabular Combined Notation (TTCN) according to TR 101 666 [27].

The ATSs were developed on a separate TTCN software tool and therefore the TTCN tables are not completely referenced in the table of contents. Each ATS contains a test suite overview part which provides additional information and references.

A.1 Version of specifications

The table below shows the version of the specifications on which the delivered ATSs are based.

Table 1: Versions of the test and Core specifications

Test specifications	TS 34.123-1 v320
	TS 34.123-2 v320
	TS 34.108 v320
	TS 34.109 v320
Core specifications	TS 25.322 v350
	TS 25.331 v350
	TS 24.008 v350
	TS 27.001 v360
	TS 22.001 v320
	TS 27.007 v360
	TS 33.102 v370
	TS 23.038 v330
	TS 23.040 v350
	TS 23.041 v330
	TS 24.011 v340

A.2 NAS ATS

The approved NAS test cases are listed.

Table A 1: NAS TTCN test cases

Test case	Description
	MM
9.1	TMSI reallocation
9.2.1	Authentication accepted
9.2.2	Authentication rejected by the network
9.2.3	Authentication rejected by the UE (MAC code failure)
9.2.4	Authentication rejected by the UE (SQN failure)
9.3.1	General Identification
9.4.1	Location updating / accepted
9.4.2.1	Location updating / rejected / IMSI invalid
9.4.2.2.1	Location updating / rejected / PLMN not allowed – test 1
9.4.2.2.2	Location updating / rejected / PLMN not allowed – test 2
9.4.2.3	Location updating / rejected / location area not allowed
9.4.2.4.1	Location updating rejected / roaming not allowed in this location area – procedure 1
9.4.2.4.2	Location updating rejected / roaming not allowed in this location area – procedure 2
9.4.2.4.3	Location updating rejected / roaming not allowed in this location area – procedure 3
9.4.2.4.4	Location updating rejected / roaming not allowed in this location area – procedure 4
9.4.2.4.5	Location updating rejected / roaming not allowed in this location area – procedure 5
9.4.3.1	Location updating / abnormal cases / random access fails
9.4.4	Location updating / release / expiry of T3240
9.4.5.1	Location updating / periodic spread
9.4.5.2	Location updating / periodic normal / test 1
9.4.5.3	Location updating / periodic normal / test 2
9.4.5.4.1	Location Updating / Periodic HPLMN search / UE waits time T.
9.4.5.4.2	Location Updating / Periodic HPLMN search / UE in manual mode
9.4.5.4.3	Location Updating / Periodic HPLMN search / UE waits at least two minutes and at most T minutes
9.4.6	Location updating / interworking of attach and periodic
9.5.2	MM connection / establishment with cipher
9.5.3	MM connection / establishment without cipher
9.5.4	MM connection / establishment rejected
9.5.5	MM connection / establishment rejected cause 4
9.5.6	MM connection / expiry T3230
9.5.7.1	MM connection / abortion by the network / cause #6
9.5.7.2	MM connection / abortion by the network / cause not equal to #6
9.5.8.1	MM connection / follow-on request pending / test 1
9.5.8.2	MM connection / follow-on request pending / test 2
9.5.8.3	MM connection / follow-on request pending / test 3
	CC
10.1.2.1.1	Outgoing call / U0 null state / MM connection requested
10.1.2.2.1	Outgoing call / U0.1 MM connection pending / CM service rejected
10.1.2.2.2	Outgoing call / U0.1 MM connection pending / CM service accepted
10.1.2.2.3	Outgoing call / U0.1 MM connection pending / lower layer failure
10.1.2.3.1	Outgoing call / U1 call initiated / receiving CALL PROCEEDING
10.1.2.3.2	Outgoing call / U1 call initiated / rejecting with RELEASE COMPLETE
10.1.2.3.3	Outgoing call / U1 call initiated / T303 expiry
10.1.2.3.4	Outgoing call / U1 call initiated / lower layer failure
10.1.2.3.5	Outgoing call / U1 call initiated / receiving ALERTING
10.1.2.3.6	Outgoing call / U1 call initiated / entering state U10
10.1.2.3.7	Outgoing call / U1 call initiated / unknown message received
10.1.2.4.1	Outgoing call / U3 UE originating call proceeding / ALERTING received
10.1.2.4.2	Outgoing call / U3 UE originating call proceeding / CONNECT received
10.1.2.4.3	Outgoing call / U3 UE originating call proceeding / PROGRESS received without in band information

10.1.2.4.4	Outgoing call / U3 UE originating call proceeding / / PROGRESS with in band information
10.1.2.4.5	Outgoing call / U3 UE originating call proceeding / DISCONNECT with in band tones
10.1.2.4.6	Outgoing call / U3 UE originating call proceeding / DISCONNECT without in band tones
10.1.2.4.7	Outgoing call / U3 UE originating call proceeding / RELEASE received
10.1.2.4.8	Outgoing call / U3 UE originating call proceeding / termination requested by the user
10.1.2.4.9	Outgoing call / U3 UE originating call proceeding / traffic channel allocation
10.1.2.4.10	Outgoing call / U3 UE originating call proceeding / timer T310 time-out
10.1.2.4.11	Outgoing call / U3 UE originating call proceeding / lower layer failure
10.1.2.4.12	Outgoing call / U3 UE originating call proceeding / unknown message received
10.1.2.4.13	Outgoing call / U3 UE originating call proceeding / CONNECT received
10.1.2.5.1	Outgoing call / U4 call delivered / CONNECT received
10.1.2.5.2	Outgoing call / U4 call delivered / termination requested by the user
10.1.2.5.3	Outgoing call / U4 call delivered / DISCONNECT with in band tones
10.1.2.5.4	Outgoing call / U4 call delivered / DISCONNECT without in band tones
10.1.2.5.5	Outgoing call / U4 call delivered / RELEASE received
10.1.2.5.6	Outgoing call / U4 call delivered / lower layer failure
10.1.2.5.7	Outgoing call / U4 call delivered / traffic channel allocation
10.1.2.5.8	Outgoing call / U4 call delivered / unknown message received
10.1.2.6.1	U10 call active / termination requested by the user
10.1.2.7.1	U11 disconnect request / clear collision
10.1.2.7.2	U11 disconnect request / RELEASE received
10.1.2.7.3	U11 disconnect request / timer T305 time-out
10.1.2.7.4	U11 disconnect request / lower layer failure
10.1.2.7.5	U11 disconnect request / unknown message received
10.1.2.8.1	U12 disconnect indication / call releasing requested by the user
10.1.2.8.2	U12 disconnect indication / RELEASE received
10.1.2.8.3	U12 disconnect indication / lower layer failure
10.1.2.8.4	U12 disconnect indication / unknown message received
10.1.2.9.1	Outgoing call / U19 release request / timer T308 time-out
10.1.2.9.2	Outgoing call / U19 release request / 2nd timer T308 time-out
10.1.2.9.3	Outgoing call / U19 release request / RELEASE received
10.1.2.9.4	Outgoing call / U19 release request / RELEASE COMPLETE received
10.1.2.9.5	Outgoing call / U19 release request / lower layer failure
10.1.3.1.1	Incoming call / U0 null state / SETUP received with a non supported bearer capability
10.1.3.2.1	Incoming call / U6 call present / automatic call rejection
10.1.3.3.1	Incoming call / U9 mobile terminating call confirmed / alerting or immediate connecting
10.1.3.3.2	Incoming call / U9 mobile terminating call confirmed / DTCH assignment
10.1.3.3.3	Incoming call / U9 mobile terminating call confirmed / termination requested by the user
10.1.3.3.4	Incoming call / U9 mobile terminating call confirmed / DISCONNECT received
10.1.3.3.5	Incoming call / U9 mobile terminating call confirmed / RELEASE received
10.1.3.3.6	Incoming call / U9 mobile terminating call confirmed / lower layer failure
10.1.3.3.7	Incoming call / U9 mobile terminating call confirmed / unknown message received
10.1.3.4.1	Incoming call / U7 call received / call accepted
10.1.3.4.2	Incoming call / U7 call received / termination requested by the user
10.1.3.4.3	Incoming call / U7 call received / DISCONNECT received
10.1.3.4.4	Incoming call / U7 call received / RELEASE received
10.1.3.4.5	Incoming call / U7 call received / lower layer failure
10.1.3.4.6	Incoming call / U7 call received / DTCH assignment
10.1.3.4.7	Incoming call / U7 call received / unknown message received
10.1.3.4.8	Incoming call / U7 call received / RELEASE COMPLETE received
10.1.3.5.1	Incoming call / U8 connect request / CONNECT acknowledged
10.1.3.5.2	Incoming call / U8 connect request / timer T313 time-out
10.1.3.5.3	Incoming call / U8 connect request / termination requested by the user

10.1.3.5.4	Incoming call / U8 connect request / DISCONNECT received with in-band information
10.1.3.5.5	Incoming call / U8 connect request / DISCONNECT received without in-band information
10.1.3.5.6	Incoming call / U8 connect request / RELEASE received
10.1.3.5.7	Incoming call / U8 connect request / lower layer failure
10.1.3.5.8	Incoming call / U8 connect request / DTCH assignment
10.1.3.5.9	Incoming call / U8 connect request / unknown message received
10.1.4.1.1	In-call functions / DTMF information transfer / basic procedures
10.1.4.2.1	In-call functions / User notification / UE terminated
10.3	User to user signalling
	General Tests
13.2.1.1	Emergency call / with USIM / accept case
13.2.2.1	Emergency call / without USIM / accept case
13.2.2.2	Emergency call / without USIM / reject case
	User Equipment features
17.1.2	Test of autocalling restrictions/ Constraining the access to a single number (TS 22.001 category 3)
17.1.3	Test of autocalling restrictions/ Constraining the access to a single number (TS 22.001 category 1 and 2)
17.1.4	Test of autocalling restrictions/ Behaviour of the UE when its list of blacklisted numbers is full

A.2.1 The TTCN Graphical form (TTCN.GR)

The TTCN.GR representation of this ATS is contained in an Adobe Portable Document Format™ file (nas.PDF contained in archive nas.ZIP) which accompanies the present document.

A.2.2 The TTCN Machine Processable form (TTCN.MP)

The TTCN.MP representation corresponding to this ATS is contained in an ASCII file (nas.MP contained in archive nas.ZIP) which accompanies the present document.

A.3 SMS ATS

The approved SMS test cases are listed.

Table A 2: NAS TTCN test cases

Test case	Description
16.1.1	SMS point to point on CS mode / SMS mobile terminated
16.1.2	SMS point to point on CS mode / SMS mobile originated
16.1.3	SMS point to point on CS mode / Test of memory full condition and memory available notification
16.1.4	SMS point to point on CS mode / Test of the status report capabilities and of SMS-COMMAND
16.1.5.1	SMS point to point on CS mode / Test of class 0 short messages
16.1.5.2	SMS point to point on CS mode / Test of class 1 short messages
16.1.5.3	SMS point to point on CS mode / Test of class 2 short messages
16.1.7	SMS point to point on CS mode / Test of the replace mechanism for SM type 1-7
16.1.8	SMS point to point on CS mode / Test of the reply path scheme
16.1.9.1	SMS point to point on CS mode / Multiple SMS mobile originated / UE in idle mode
16.1.9.2	SMS point to point on CS mode / Multiple SMS mobile originated / UE in active mode
16.3	Short message service cell broadcast

A.3.1 The TTCN Graphical form (TTCN.GR)

The TTCN.GR representation of this ATS is contained in an Adobe Portable Document Format™ file (sms.PDF contained in archive sms.ZIP) which accompanies the present document.

A.3.2 The TTCN Machine Processable form (TTCN.MP)

The TTCN.MP representation corresponding to this ATS is contained in an ASCII file (sms.MP contained in archive sms.ZIP) which accompanies the present document.

A.4 RRC ATS

The approved RRC test cases are listed.

Table A 3: RRC TTCN test cases

Test case	Description
8.1.1.1	Paging for Connection in idle mode
8.1.1.2	Paging for Connection in connected mode(CELL_DCH)
8.1.1.3	Paging for Connection in connected mode(URA_PCH)
8.1.1.4	Paging for Notification in idle mode
8.1.1.5	Paging for Notification in connected mode (CELL_PCH)
8.1.1.6	Paging for Notification in connected mode (URA_PCH)
8.1.1.7	Paging for Notification in connected mode (CELL_DCH)
8.1.1.8	Paging for Notification in connected mode (CELL_FACH)
8.1.2.1	RRC Connection Establishment in CELL_DCH state: Success
8.1.2.2	RRC Connection Establishment: Success after T300 timeout
8.1.2.3	RRC Connection Establishment: Failure (V300 is greater than N300)
8.1.2.4	RRC Connection Establishment: Reject (“wait time” is not equal to 0)
8.1.2.5	RRC Connection Establishment: Reject (“wait time” is not equal to 0 and V300 is greater than N300)
8.1.2.6	RRC Connection Establishment: Reject (“wait time” is set to 0)
8.1.2.7	RRC Connection Establishment in CELL_FACH state: Success
8.1.2.8	RRC Connection Establishment : Invalid system information message reception
8.1.3.1	RRC Connection Release in CELL_DCH state: Success
8.1.3.2	RRC Connection Release in CELL_FACH state: Success
8.1.3.3	RRC Connection Release using on CCCH in CELL_FACH state: Success
8.1.3.4	RRC Connection Release in CELL_FACH state: Failure
8.1.3.5	RRC Connection Release in CELL_FACH state: Invalid message
8.1.5.1	UE Capability: Success
8.1.5.2	UE Capability: Success after T304 timeout
8.1.5.3	UE Capability in CELL_DCH state: Failure (After N304 retransmissions)
8.1.5.4	UE Capability in CELL_DCH state: Success
8.1.5.5	UE Capability in CELL_FACH state: Success after T304 timeout
8.1.6.1	Direct Transfer in CELL DCH state(invalid message reception)
8.1.6.2	Direct Transfer in CELL FACH state(invalid message reception)
8.1.7.1	Security mode control in CELL_DCH state
8.1.7.2	Security mode control in CELL_FACH state
8.1.8.1	Counter check in CELL_DCH state
8.1.8.2	Counter check in CELL_FACh state
8.1.9.1	Signalling Connection Release Request
8.3.3.1	RNTI reallocation: Success
8.3.3.2	RNTI reallocation: Failure (Invalid message reception)
8.3.1.1	Cell Update: cell reselection in CELL_FACH
8.3.1.2	Cell Update: cell reselection in CELL_PCH
8.3.1.3	Cell Update: periodical cell update in CELL_FACH
8.3.1.4	Cell Update: periodical cell update in CELL_PCH
8.3.1.5	Cell Update: UL data transmission in URA_PCH
8.3.1.6	Cell Update: UL data transmission in CELL_PCH
8.3.1.7	Cell Update: paging response in URA_PCH
8.3.1.8	Cell Update: Paging response in Cell_PCH
8.3.1.9	Cell Update: re-entering of service area after T305 expiry and being out of service area
8.3.1.10	Cell Update: expiry of T307 after T305 expiry and being out of service area
8.3.1.11	Cell Update: Success after T302 time-out
8.3.1.12	Cell Update: Failure (After Maxium Re-transmissions)
8.3.1.13	Cell Update: Reception of Invalid CELL UPDATE CONFIRM Message
8.3.1.14	Cell Update: Radio Bearer Control for Transition from CELL_DCH to CELL_FACH
8.3.1.15	Cell Update: Acknowledged Mode RLC Reset
8.3.1.16	Cell Update: cell reselection in CELL_FACH
8.3.1.17	Cell Update: Failure (UTRAN initiate an RRC connection release procedure on DCCH)
8.3.2.1	URA Update: URA reselection
8.3.2.2	URA Update: periodical URA update

8.3.2.3	URA Update: re-entering of service area after T306 expiry
8.3.2.4	URA Update: loss of service after expiry of timers T307 and T306
8.3.2.5	URA Update: Success After Confirmation error of URA-ID list
8.3.2.6	URA Update: Failure(V303 is greater than N303: Confirmation error of URA-ID list)
8.3.2.7	URA Update: Success after T303 timeout
8.3.2.8	URA Update: Failure(V303 is greater than N303: T303 timeout)
8.3.2.9	URA Update: Failure(UTRAN initiate an RRC connection release procedure on DCCH)
8.3.4.1	Active set update in soft handover: Radio Link Addition
8.3.4.2	Active set update in soft handover: Radio Link Removal
8.3.4.3	Active set update in soft handover:Combined radio link addition and removal (Active set is not full)
8.3.4.4	Active set update in soft handover: Unsupported configuration in te UE
8.3.4.5	Active set update in soft handover:Combined radio link addition and removal (Active set is full)
8.3.4.6	Active set update in soft handover:Subsequent reception of ACTIVE SET UPDATE message/Incompatible simultaneous reconfiguration
8.3.4.7	Active set update in soft handover: Invalid Message Reception

A.4.1 The TTCN Graphical form (TTCN.GR)

The TTCN.GR representation of this ATS is contained in an Adobe Portable Document Format™ file (rrc.PDF contained in archive rrc.ZIP) which accompanies the present document.

A.4.2 The TTCN Machine Processable form (TTCN.MP)

The TTCN.MP representation corresponding to this ATS is contained in an ASCII file (rrc.MP contained in archive rrc.ZIP) which accompanies the present document.

A.5 RLC ATS

. The approved RRC test cases are listed.

Table A 4: RLC TTCN test cases

Test case	Description
7.2.2.2	UM Segmentation and reassembly / Selection of 7 or 15 bit Length Indicators
7.2.2.3	UM Segmentation / 7-bit Length Indicators / Padding
7.2.2.4	UM Segmentation / 7-bit Length Indicators / LI = 0
7.2.2.5	UM Segmentation / 7-bit Length Indicators / Invalid LI value
7.2.2.6	UM Segmentation / 7-bit Length Indicators / LI value > PDU size
7.2.2.7	UM Segmentation / 7-bit Length Indicators / First data octet LI
7.2.2.8	UM Segmentation / 15-bit Length Indicators / Padding
7.2.2.9	UM Segmentation / 15-bit Length Indicators / LI = 0
7.2.2.10	UM Segmentation / 15-bit Length Indicators / One octet short LI
7.2.2.11	UM Segmentation / 15-bit Length Indicators / LI value > PDU size
7.2.2.12	UM Segmentation / 15-bit Length Indicators / First data octet LI
7.2.3.2	AM Segmentation and reassembly / Selection of 7 or 15 bit Length Indicators
7.2.3.3	AM Segmentation / 7-bit Length Indicators / Padding or Piggy-backed Status
7.2.3.4	AM Segmentation / 7-bit Length Indicators / LI = 0
7.2.3.5	AM Segmentation / 7-bit Length Indicators / Reserved LI value
7.2.3.6	AM Segmentation / 7-bit Length Indicators / LI value > PDU size
7.2.3.7	AM Segmentation / 15-bit Length Indicators / Padding or Piggy-backed Status
7.2.3.8	AM Segmentation / 15-bit Length Indicators / LI = 0
7.2.3.9	AM Segmentation / 15-bit Length Indicators / One octet short LI
7.2.3.10	AM Segmentation / 15-bit Length Indicators / Reserved LI value
7.2.3.11	AM Segmentation / 15-bit Length Indicators / LI value > PDU size
7.2.3.12	AM Correct use of Sequence Numbering
7.2.3.13	AM Control of Transmit Window
7.2.3.14	AM Control of Receive Window
7.2.3.15	AM Polling for status / Last PU in transmission queue
7.2.3.16	Polling for status / Last PU in retransmission queue
7.2.3.17	AM Polling for status / Poll every Poll_PU Pus
7.2.3.25	AM Receiver Status Triggers / Detection of missing Pus
7.2.3.28	AM Status reporting / Abnormal conditions / Reception of LIST SUFI with Length set to zero
7.2.3.30	AM SDU discard after MaxDAT number of retransmissions

A.5.1 The TTCN Graphical form (TTCN.GR)

The TTCN.GR representation of this ATS is contained in an Adobe Portable Document Format™ file (rlc.PDF contained in archive rlc.ZIP) which accompanies the present document.

A.5.2 The TTCN Machine Processable form (TTCN.MP)

The TTCN.MP representation corresponding to this ATS is contained in an ASCII file (rlc.MP contained in archive rlc.ZIP) which accompanies the present document.

A.6 MAC ATS

tbd

A.7 BMC ATS

tbd

A.8 PDCP ATS

Tbd

Annex B (normative): Partial IXIT proforma

B.0 Introduction

This partial IXIT proforma contained in the present document is provided for completion, when the related Abstract Test Suite is to be used against the Implementation Under Test (IUT).

Text in *italics* is comments for guidance for the production of a IXIT, and is not to be included in the actual IXIT.

The completed partial IXIT will normally be used in conjunction with the completed ICS, as it adds precision to the information provided by the ICS.

B.1 Parameter values

B.1.1 L2M Test Suite Parameter Declarations

The following parameters are common to all ATSs.

Table B1: L2M PIXIT

Parameter Name	Description	Type	Default Value	Supported Value
px_CRNTI	C RNTI	C_RNTI	'0000 0000 0000 0001'B	
px_CodeNumAICH	Channelization code number for AICH (spreading factor is 256), default value = 255 (from default SIB5)	INTEGER	255	
px_CodeNumberPICH	Channelization code for PICH, the speading factor is 256, default value = 255 (from default SIB5)	ChannelisationCode256	255	
px_DL_TxPower_DPCH_34_StandAlone	Downlink transmission power level of DPCH used for 3.4 kbps stand-alone SRB	DL_TxPower	5	
px_DL_TxPower_DPCH_64K	Down link transmission power level of DPCH used for 64 kbps PS SRB	DL_TxPower	5	
px_DL_TxPower_DPCH_122AMR	Down link transmit power level of DPCH used for 12.2 kbps AMR	DL_TxPower	5	
px_PowerAICH	Transmission power level of AICH	DL_TxPower	-5	
px_PowerPICH	Transmission power level of primary PICH	DL_TxPower	-5	
px_PowerpCCPCH	Transmission power level of primary CCPCH	DL_TxPower	-2	
px_PowerpSCH	Transmission power level of primary SCH	DL_TxPower	-5	
px_PowersSCH	Transmission power level of secondary SCH	DL_TxPower	-5	
px_PowersCCPCH1	Transmission power level of secondary CCPCH1	DL_TxPower	-2	
px_SlotFormatsCCPCH1	Channelization code for secondary CCPCH1 when speading factor = 64	SCCPCHSlotFormat	4	
px_TimingsCCPCH1	Timing offset for secondary CCPCH1	INTEGER	0	
px_SndScrmCodeAICH	Secondary scrambling code for AICH channel, default value = 1(from default SIB5)	INTEGER	1	
px_SndScrmCodePICH	Secondary scrambling code for PICH channel, default value =2 (from default SIB5)	INTEGER	2	
px_SprdFct	Spread Fact of uplink DPCH info, see constraint " cb_Ul_DpchInfo"	SpreadingFactor	sf16	
px_SRNC_Id	SRNC Id	SRNC_Identity	'0000 0000 0001'B	
px_SRNC_IdDiff	Different value for SRNC Id than in px_SRNCId	SRNC_Identity	'0000 0000 0010'B	
px_SRNTI	S RNTI	S_RNTI	'0000 0000 0000 0000 0001'B	
px_SRNTI_Diff	Different value for S RNTI than in px_SRNTI	S_RNTI	'0000 0000 0000 0000 0010'B	
px_SF512	Spread Factor and Code Number in downlink DPCH information per radio link code list, see constraint "cb_DL_InformationPerRL"	SF512_AndCodeNumber	sf16	
px_IMSI_Diff	Different IMSI from the IMSI stored in the USIM	HEXSTRING	'001010654321063'H	
px_MA_Code	Message Authentication Code (32) MAC-I	MessageAuthenticationCode	See note 1	

px_IntegrityProtectInitNumber	Integrity Protect Initial Number	IntegrityProtInitNumber	See note 1	
px_PuncLimit	Puncturing limit for PRACH	PuncturingLimit	Pl1	
px_RB_ConfigType	Type of Radio Access Bearer to be established	RB_ConfigType	DCCH_3_4k	
px_SF_PRACH	Speading factor for PRACH	SF_PRACH	Sfpr64	
px_PowerpCPICH_A	Transmission power level of primary CPICH (for the default Cell A)	DL_TxPower_PCPICH	-60	
px_UARFCN_U_A	Uplink UARFCN number, default value = 9750 (for the default Cell:A)	INTEGER	9750	
px_UARFCN_D_A	Default value = px_UARFCNu + 950 (for the default Cell A)	INTEGER	10700	
px_PriScrmCodeA	Primary scrambling code (for the default Cell A)	PrimaryScramblingCode	100	
px_TcellA	Timing offset between BFN and SCH (for the default Cell A)	Tcell	0	
px_CipheringOnOff	Security mode - TRUE if ciphering is applicable	BOOLEAN	FALSE	
px_IntegrityOnOff	Integrity mode – TRUE if integrity is applicable	BOOLEAN	FALSE	
px_RAT	This parameter is used to specify which radio access technology is being used for the current test execution. Valid values: fdd and tdd	RatType	fdd	
px_AuthAMF	Authentication Management Field (16 bits)	BITSTRING	See note 2	
px_AuthK	Authentication Key (128 bits)	BITSTRING	'0101111001001 0101011001101 0110001001000 1001101110101 1101001010101 1101110100000 0100101110011 0011111000011 0000100110100 11000101001'B	
px_AuthRAND	Random Challenge (128 bits)	BITSTRING	'01010101…01'B	
px_IMEI_Def	Default IMEI value	HEXSTRING	See note 1	
px_IMEISV_Def	Default IMEISV value	HEXSTRING	See note 1	
px_IMSI_Def	Default IMSI value	HEXSTRING	'001010123456063'H	
px_TMSI_Def	Default TMSI	OCTETSTRING	'12345678'O	
px_CiphAlgCap	Ciphering Algorithm Capability	B16	'0000000000000001'B	
px_KeySeqDef	Default Key Sequence	Keyseq	'101'B	
px_MSClsmk_A5_1	Default Algorithm A5/1 supported	B1	'0'B	
px_MSClsmk_ESIND	Default Early Sending Indication	B1	'0'B	
px_MSClsmk_RF_PwrCap	Default RF Power Capability	B3	'000'B	
px_MSClsmk_RevLvl	Default Revision Level	B2	'10'B	
px_Auth_n	Value of n to initialize tcv_Auth_n (length of extended response) min 31, max 127 (TS 34.108 cl. 8.1.2)	INTEGER	31	
px_FRESH	Value for FRESH	Fresh	See note 1	
px_CipheringAlgorithm	Ciphering Algorithm uea0 - 0, uea1 - 1, ...	CipheringAlgorithm	0	
px_CS	TRUE if Circuit Switched is applicable	BOOLEAN	TRUE	
px_PS	TRUE if Packet Switched is applicable	BOOLEAN	TRUE	
px_NMO	This parameter is used to specify network operation mode. Valid values: '00'O and '01'O	OCTETSTRING	'00'O	
px_RAC	This parameter is used to specify	OCTETSTRI		

	routing arear code, valid length of the octetstring is 2.	NG		
px_Region	This parameter is used to specify for which region the system information blocks are broadcast in the test execution. Valid values: "Japan" and "Europe".	Region	"Europe"	
px_SF_Pilot	Spread Fact and Pilot of Downlink information common for all radio links	SF512_AndPilot	sfd512	
NOTE 1: No default value can be proposed (Manufacturer defined value). NOTE 2: No default value can be proposed, because not enough information is available in TS34.109 clause 8.1.2.				

B.1.2 L3M Test Suite Parameters Declarations

The following parameters are commonly used in the RRC and NAS ATSs.

Table B2: L3M PIXIT

Parameter Name	Description	Type	Default Value	Supported Value
px_MSClsmk_CM3	Default Classmark 3 Indicator	B1	'0'B	
px_MSClsmk_CMSP	Default CM Service Prompt Support	B1	'0'B	
px_MSClsmk_A5_2	Default Algorithm A5/2 supported	B1	'0'B	
px_MSClsmk_A5_3	Default Algorithm A5/3 supported	B1	'0'B	
px_MSClsmk_FreqCap	Default Frequency Capability	B1	'0'B	
px_MSClsmk_LCSVA_Cap	Default LCSVA Capabilities Support	B1	'0'B	
px_MSClsmk_PS_Cap	Default Pseudo Synchronisation Capability	B1	'0'B	
px_MSClsmk_SoLSA	Default SoLSA supported	B1	'0'B	
px_MSClsmk_SM_Cap	Default Short Message Capability	B1	'1'B	
px_MSClsmk_SSSI	Default SS Screen Indicator	B2	'01'B	
px_MSClsmk_UCS2	Default UCS2 encoding supported	B1	'0'B	
px_MSClsmk_VBS	Default VBS Capability	B1	'0'B	
px_MSClsmk_VGCS	Default VGCS Capability	B1	'0'B	
px_CC_Serv	Service selected for Mobile Originated calls and Mobile Terminated calls. The possible values are ("Telephony", "EmergencyCall", "31kHz", "V110", "V120", "PIAFS", "FTM", "X31", "BTM", "MmediaCall")	Services	"31kHz"	
px_BcapDataCompression	Data compression supported (used in the Bearer Capability)	B1	'0'B	
px_BcapFNUR	Fixed Network User rate supported: '00001'B: FNUR 9.6 kbit/s '00010'B: FNUR 14.4 kbit/s '00011'B: FNUR 19.2 kbit/s '00100'B: FNUR 28.8 kbit/s '00101'B: FNUR 38.4 kbit/s '00110'B: FNUR 48.0 kbit/s '00111'B: FNUR 56.0 kbit/s '01000'B: FNUR 64.0 kbit/s '01001'B: FNUR 33.6 kbit/s '01010'B: FNUR 32.0 kbit/s	B5	'00001'B	
px_BcapITC	Information transfer capability supported (used for the generation of the Bearer Capability) 0 - UDI 1 - RDI 2 - 31 kHz Audio 3 - Other	ItcInt	2	
px_BcapModemType	Modem type supported (used in the Bearer Capability)	B5	'00110'B	
px_BcapNumberDataBits	Number of data bits supported (used in the Bearer Capability)	B1	8	
px_BcapNumberStopBits	Number of Stops bits supported (used in the Bearer Capability)	B1	2	
px_BcapOtherModemType	Other modem type supported (used in the Bearer Capability)	B2	'10'B	
px_BcapParity	Parity supported (used in the Bearer Capability)	B3	'011'B	
px_BcapSACP	Signalling access protocol supported (used in the Bearer Capability)	B3	'001'B	
px_BcapSyncAsync	Synchronous '0'B or Asynchronous '1'B mode supported by IUT	B1	'1'B	
px_BcapUeFlowControl	UE flow control. 0-outband, 1-inband, 2-no flow control. 3- X.25 4- X.75	FlowControl	0	

	Default: 0, outband flow control			
px_CC_CallDiallingDigits	Dialling digits used to initiate a CC MO call	IA5String	"0123456902"	
px_UARFCNU_B	RF frequency number for uplink Cell B	INTEGER	9700	
px_UARFCN_D_B	RF frequency number for downlink Cell B	INTEGER	10650	
px_PriScrmCodeB	Primary scrambling code for Cell B	PrimaryScramblingCode	150	
px_TcellB	Timing offset between BFN and SCH Cell B	Tcell	0	

B.1.3 NAS Test Suite Parameters Declarations

The following parameters are commonly used in the NAS ATS.

Table B3: NAS PIXIT

Parameter Name	Description	Type	Default Value	Supported Value
px_CC_ServNotSupp	Not supported service selected for Mobile Originated calls and Mobile Terminated calls. The possible values are ("Telephony", "EmergencyCall", "31kHz", "V110", "V120", "PIAFS", "FTM", "X31", "BTM", "MmediaCall")	Services	"Alternate Speech/Fascimile"	
px_DTMF_BasicCharSet	TRUE if DMTF Chars 0-9, *, # supported	BOOLEAN	TRUE	
px_DTMF_ToneInd	TRUE if UE support DTMF tone indication	BOOLEAN	TRUE	
px_DTMF_OtherCharSet	TRUE if DMTF Chars A, B, C, D supported	BOOLEAN	TRUE	
px_UuInfo	User-user information for TC 10_3	OCTETSTRING	''O	
px_Uupd	User-user protocol discriminator for TC 10_3	B8	'00000100'B	
px_AuthRAND_2	A second Random Challenge (128 bits)	BITSTRING	'1010101…10'B	
px_UARFCN_U_C	RF frequency number for uplink Cell C	INTEGER	9800	
px_UARFCN_D_C	RF frequency number for downlink Cell C	INTEGER	10750	
px_PriScrmCodeC	Primary scrambling code for Cell C	PrimaryScramblingCode	200	
px_TcellC	Timing offset between BFN and SCH Cell C	Tcell	0	
px_MaxRetrans	maximum number of RRC CONNECTION request retransmissions	INTEGER	4	
px_TMSI2	Second TMSI value	OCTETSTRING	'09876543'O	
px_KeySeq2	Second key sequence	KeySeq	'000'B	
px_AutocallingBlacklistNumber	Number of B-party numbers that can be stored in the list of blacklisted numbers	INTEGER	20	
px_AutocallingNumber	Called number to be used for autocalling	IA5String	"0613454120"	
px_AutocallingCause1or2	Cause value of categorie 1 or 2 to be used in TC_17_1_3	INTEGER	18	
px_AutocallingRepeatCat1or2	Number of repeat attempt done for the categorie 1 or 2 to be used in TC_17_1_3	INTEGER	10	

B.1.4 SMS Test Suite Parameters Declarations

These parameters are used in the SMS ATS.

Table B4: SMS PIXIT

Parameter Name	Description	Type	Default Value	Supported Value
px_TC1M	Value for timer TC1M, to be declared by the manufacturer	INTEGER	10000	
px_MaxCP_DataRetx	max. number of CP data retransmissions for SMS	INTEGER	3	

B.1.5 MMI questions

The table below shows the list of MMI questions used in the ATSs, the column 'ATS' indicates in which ATS the question is used.

Table B5: MMI questions

MMI question	ATS
How to switch the PLMN selection mode of the UE to automatic selection? NOTE: Used in the step ts_UT_PLMN_SelModeAuto	All ATSs
How to switch the PLMN selection mode of the UE to manual selection? NOTE: Used in the step ts_UT_PLMN_SelModeMan	All ATSs
How to select a given PLMN manually? NOTE: Used in the step ts_UT_PLMN_SelPerf	All ATSs
How to power off the UE? NOTE: Used in the step ts_UT_UE_PwrOff	All ATSs
How to power on the UE? NOTE: Used in the step ts_UT_UE_PwrOn	All ATSs

How to switch off the UE? NOTE: Used in the step ts_UT_UE_SwitchOff	All ATSs
How to switch on the UE? NOTE: Used in the step ts_UT_UE_SwitchOn	All ATSs
How to insert the USIM card into the UE? NOTE: Used in the step ts_UT_USIM_Insert	All ATSs
How to remove the USIM card from the UE? NOTE: Used in the step ts_UT_USIM_Remove	All ATSs
How to check that DTCH is trough connected ? NOTE: Used in steps ts_UT_CheckDTCH_ThroughConnected and ts_UT_DTCH_NotthroughConnected	RRC, SMS, NAS
How to configure UE for a MO telephony call? NOTE: Used in the step ts_UT_ConfigUE_MO	RRC, SMS, NAS
How to configure UE for an emergency call? NOTE: Used in the step ts_UT_ConfigUE_MO	RRC, SMS, NAS
How to configure UE for a MT telephony call?	RRC, SMS, NAS

NOTE: Used in the step ts_UT_ConfigUE_MT	
How to send any NAS message in order for RRC to receive data? NOTE: Used in the step ts_UT_SendData	RRC, SMS, NAS
How to initiate a non call related supplementary service which is supported by the UE? NOTE: Used in the step ts_UT_InitNonCallRelSS	NAS
How to initiate sending of a mobile originated short message from the UE? NOTE: Used in the step ts_UT_InitSMS_MO	NAS
How to insert 2nd SIM card with short IMSI? NOTE: Used in the step ts_UT_USIM2_Insert	NAS
How to initiate an autocalling call with a given number? NOTE: Used in the step ts_UT_AutocallingInitCall	NAS
How to initiate an autocalling call for a number that will be put in the blacklisted list? NOTE: Used in the step ts_UT_AutocallingInitCall_BlackList	NAS
How to reset the autocalling list of blacklisted numbers? NOTE: Used in the step ts_UT_AutocallingresetBlacklist	NAS

How to check that the DTMF tone indication has been generated? NOTE: Used in the step ts_UT_DTMF_CheckTone	NAS
How to enable call refusal on the UE? NOTE: Used in the step ts_UT_EnableCallRefusal	NAS
How to check the contents of the received CBS? NOTE: Used in the step ts_UT_ChkCBS_Received	SMS
How to check that the Memory Capacity Exceeded Flag has been set to the USIM simulator? NOTE: Used in the step ts_UT_ChkMCEF_Set	SMS
How to check if the Memory Capacity Exceeded Flag has been unset on the USIM simulator? NOTE: Used in the step ts_UT_ChkMCEF_Unset	SMS
How to check the length and the contents of a given received Short Message ? NOTE: Used in the step ts_UT_ChkMsgDisplayed	SMS
How to check whether the USIM simulator indicated an attempt made by the ME to store the short message in the USIM and return the status response 'Memory Problem'('92 40')? NOTE: Used in the step ts_ChkUSIM_AttIndMemProblem	SMS
How to check whether the USIM simulator indicates an attempt made by the ME to store the short message in the USIM and returns the status response 'OK' ('90 00')?	SMS

NOTE: Used in the step ts_UT_ChkUSIM_AttIndOK	
How to connect the USIM simulator to the UE? NOTE: Used in the step ts_UT_ConnectUSIM_Simulator	SMS
How to send an SMS COMMAND message containing a request to delete the previously submitted Short Message? NOTE: Used in the step ts_UT_InitSMS_CmdDel	SMS
How to send an SMS COMMAND message containing an enquiry about the previously submitted SM? NOTE: Used in the step ts_UT_InitSMS_CmdEnq	SMS
How to check that NO recalled short Message is displayed? NOTE: Used in the step ts_UT_RecallAndNotDisplay	SMS
How to reply to a short Message with a given length? NOTE: Used in the step ts_UT_SM_Reply	SMS

Annex C (informative): Additional information to IXIT

Additional information may be provided when completing the IXIT questions listed in annex A.

C.1 Identification Summary

This table is completed by the test laboratory. The item "Contract References" is optional.

Table C1: Identification Summary

IXIT Reference Number	
Test Laboratory Name	
Date of Issue	
Issued to (name of client)	
Contract References	

C.2 Abstract Test Suite Summary

In the following table the test laboratory provides the version number of the protocol specification and the version number of ATS which are used in the conformance testing.

Table C2: ATS Summary

Protocol Specification	**TS 25.331**
Version of Protocol Specification	
Test Specification in prose	**TS 34.123-1**
Version of TSS & TP Specification	
ATS Specification	**TS 34.123-3**
Version of ATS Specification	
Abstract Test Method	**Distributed Test Method**

C.3 Test Laboratory

C.3.1 Test Laboratory Identification

The test laboratory provides the following information.

Table C3: Test Laboratory Identification

Name of Test Laboratory	
Postal Address	
Office address	
e-mail address	
Telephone Number	
FAX Number	

C.3.2 Accreditation status of the test service

The test laboratory provides the following information.

Table C4: Accreditation status of the test service

Accreditation status	
Accreditation Reference	

C.3.3 Manager of Test Laboratory

The test laboratory provides the information about the manager of test laboratory in the following table.

Table C5: Manager of Test Laboratory

Name of Manager of Test Laboratory	
e-mail address	
Telephone Number	
FAX Number	
E-mail Address	

C.3.4 Contact person of Test Laboratory

The test laboratory provides the information about the contact person of test laboratory in the following table.

Table C6: Contact person of Test Laboratory

Name of Contact of Test Laboratory	
e-mail address	
Telephone Number	
FAX Number	
E-mail Address	

C.3.5 Means of Testing

In the table below, the test laboratory provides a statement of conformance of the Means Of Testing (MOT) to the reference standardized ATS, and identifies all restrictions for the test execution required by the MOT beyond those stated in the reference standardized ATS.

Table C7: Means of Testing

Means of Testing

C.3.6 Instructions for Completion

In this table, the test laboratory provides any specific instructions necessary for completion and return of the proforma from the client.

Table C8: Instruction for Completion

Instructions for Completion

C.4 Client

C.4.1 Client Identification

The client provides the identification in the following table.

Table C9: Client Identification

Name of Client	
Postal Address	
Office Address	
Telephone Number	
FAX Number	

C.4.2 Client Test Manager

In this table the client provides information about the test manager.

Table C10: Client Test Manager

Name of Client Test Manager	
Telephone Number	
FAX Number	
E-mail Address	

C.4.3 Client Contact person

In this table the client provides information about the test contact person.

Table C11: Client Contact person

Name of Client contact person	
Telephone Number	
FAX Number	
E-mail Address	

C.4.4 Test Facilities Required

In the following table, the client records the particular facilities required for testing, if a range of facilities is provided by the test laboratory.

Table C12: Test Facilities Required

Test Facilities Required

C.5 System Under Test

C.5.1 SUT Information

The client provides information about the SUT in the table below.

Table C13: SUT Information

System Name	
System Version	
SCS Reference	
Machine Configuration	
Operating System Identification	
IUT Identification	
ICS Reference for the IUT	

C.5.2 Limitations of the SUT

In the table below, the client provides information explaining if any of the abstract tests cannot be executed.

Table C14: Limitation of the SUT

Limitations of the SUT

C.5.3 Environmental Conditions

In the table below the client provides information about any tighter environmental conditions for the correct operation of the SUT.

Table C15: Environmental Conditions

Environmental Conditions

C.6 Ancillary Protocols

This clause is completed by the client in conjunction with the test laboratory.

In the following tables, the client identifies relevant information concerning each ancillary protocol in the SUT other than the IUT itself. One table for one ancillary protocol.

Based on the MOT the test laboratory should create question proforma for each ancillary protocol in the blank space following each table. The information required is dependent on the MOT and the SUT, and covers all the addressing, parameter values, timer values and facilities (relevant to ENs) as defined by the ICS for the ancillary protocol.

C.6.1 Ancillary Protocols 1

Table C16: Ancillary Protocol 1

Protocol Name	
Version number	
ICS Reference (optional)	
IXIT Reference (optional)	
PCTR Reference (optional)	

C.6.2 Ancillary Protocols 2

Table C17: Ancillary Protocol 2

Protocol Name	
Version number	
ICS Reference (optional)	
IXIT Reference (optional)	
PCTR Reference (optional)	

Annex D (informative): PCTR Proforma

PROTOCOL

Conformance Test Report

(PCTR)

Universal Mobile Telecommunication System, UMTS,
User Equipment-Network Access

Layer 3 Signalling Functions

Test Candidate	
Name :	SUT name
Model :	model
H/W version :	hw
S/W version :	sw
Serial No. :	serienr

Client	
Name :	
Street / No. :	
Postal Code / City:	
Country :	

Annex E (informative): TTCN style guide for 3GPP ATS

E.1 Introduction

This annex provides a set of coding standards and development guidelines for use in the development of TTCN abstract test suites for ensuring that user equipment for the 3GPP standard conforms to the relevant core specifications.

The following items are assumed to exist, but their specification is outside the scope of this annex.

- A complete unambiguous prose detailing all test cases to be implemented,
- A complete unambiguous set of core specifications,
- A complete unambiguous detailed description of all the messages that are to be sent,
- A tool or human process that can convert Test Suite Operation Definitions to physical processes within the test system or unit under test,
- An abstracted or generic application programmers interface to all hardware components in the system,
- A tool for the translation and/or compilation of ISO/IEC 9646 series TTCN to run on a test platform.

It is recognised within the context of the 3GPP User Terminal that some of these items are not yet stabilised.

The structure of the present annex maps directly to the guidelines provided in ETR 141. Rules are repeated in the present annex for convenience, with additional information specific to 3GPP test suite development provided where relevant. For more detailed information or examples about the rules, see ETR 141.

In the present annex, the terms 'should' and 'shall' are frequently used. For the purpose of this annex, the following definitions apply:

Shall means that the rule must be adhered to for all ATS development. If a rule expressed in terms of 'shall' is not followed, either the ATS must be updated so that the rule is followed, or the rule in the coding conventions must be updated to resolve the difference.

Should means that the rule is a guideline. If a rule expressed in terms of 'should' is broken, a brief comment should be provided describing why the guideline does not apply.

E.2 ETR 141 rules and applicability

RULE 1: Statement of naming conventions
Naming conventions should be explicitly stated. Naming conventions should not exist only for a single ATS, and the reader of an ATS should not be forced to "derive" the rules implicitly. The naming conventions should be part of the ATS conventions contained in the ATS specification document.

Names used in the present annex are comprised of a prefix part and a name body part. Conventions for deriving prefixes and name bodies are described after Rule 3 in the present annex.

RULE 2: Coverage of naming conventions
Naming conventions stated should, as a minimum, cover the following TTCN objects: - test suite parameters/constants/variables; - test case variables; - formal parameters; - timers; - PDU/ASP/structured types; - PDU/ASP/structured types constraints; - test suite operations; - aliases; - test case/test step identifiers.

RULE 3: General properties of naming conventions
a) Protocol standard aligned When there is a relationship between objects defined in the ATS and objects defined in the protocol standard, e.g. PDU types, the same names should be used in the ATS if this does not conflict with the character set for TTCN identifiers or with other rules. In case of a conflict, similar names should be used. **b) Distinguishing** The naming conventions should be defined in such a way, that objects of different types appearing in the same context, e.g. as constraint values, can be easily distinguished. **c) Structured** When objects of a given type allow a grouping or structuring into different classes, the names of these objects should reflect the structuring, i.e. the names should be composed of 2 or more parts, indicating the particular structure elements. **d) Self-explaining** The names should be such that the reader can understand the meaning (type/value/contents) of an object in a given context. When suffixes composed of digits are used, it is normally useful to have some rule expressed explaining the meaning of the digits. **e) Consistent** The rules stated should be used consistently throughout the document, there should be no exceptions. **f) Appropriate name length** Following the above rules extensively may occasionally lead to very long names, especially when structuring is used. The names should still be easily readable. When TTCN graphical form (TTCN.GR) is used, very long names are very inconvenient. NOTE: Also, test tools may not be able to implement very long identifier names, which is an important aspect in this context.

E.2.1 Multiple words are separated by upper case letters at the start of each word

Many names consist of more words, and it shall be easy to distinguish the different words building up the same name. For all TTCN Object classes this is done using the case of the letters.

This rule is mandatory for all names appearing in the body of a dynamic behaviour table, and is recommended for all other TTCN object classes.

Generally every word a name consists of shall start with an upper case letter and the rest of this word shall be in lower case letters.

E.g. "channel" + "description" -> "ChannelDescription"

This rule also applies if a word starts after another upper case letter.

E.g. "px" + "Cell" + "A" + "Cell" + "Id" -> px_CellACellId

This rule also applies if the name has a prefix, which is always lower case.

E.g. A test case variable "sequence" + "number" -> tcv_SequenceNumber

This rule does not apply if the word is a unit, in which case the word retains it's original case.

E.g. Power level 1.5 dBm ->PowerLvl1_5dBm

This rule does not apply if the word in the name is an acronym, in which case the word retains it's normal case.

- If an acronym is followed by another word, an underscore shall be used to separate the acronym from the following word.

  E.g.: "this" + "Is" + "SIM" + "Message" + "With" + "CC" + "And" + "RR" + "Things" + "In" + "It" -> "thisIsSIM_MessageWithCC_AndRR_ThingsInIt"

- An exception to acronyms retaining their case is if the name is a field / element / parameter in a structured type / PDU / ASP, in which case it must start with a lower case letter.

  E.g. "SCH" + "info" + "element" -> "sCH_InfoElement"

- A further exception to acronyms retaining their case is if the name is an ASN.1 constraint, in which case, in which case the first letter is upper case, and the remaining letters are lower case.

For all objects used in the body of dynamic behaviour tables, use of underscores is discouraged, except for the following situations:

- As a replacement for a '.'. E.g. Test case that maps to prose clause 7.2.3.1 -> tc_7_2_3_1
- To separate prefixes from names.
- To separate acronyms from the following word.
- To replace hyphens when types are re-used / imported from core specifications. This applies to types imported from ASN.1 definitions, and to names derived from table definitions in core specifications.
- To separate an ASP name from the embedded PDU name when the metatype PDU is not used. E.g RRC_DataInd_ConnAck for an RRC data indication ASP with an embedded CONNECT ACKNOWLEDGE PDU.

E.2.2 Identifiers shall be protocol standard aligned

To support rule 3(a), the mapping guidelines in table 0 shall be used. This mapping table also supports rule 6.

Table E1: Mapping guidelines between protocol standards and identifiers

Type	Naming rule
Objects of Structured Type	Shall be derived from the name of the Information Element in the standard, if it corresponds to this (use standard acronyms where appropriate). E.g.: "Window Size super-field" -> "WindowSizeSUFI"
Fields in a Structured Type	Shall be derived from the name of the same field in the corresponding Information Element in the standard. (Acronyms for the entire field name shall not be used) E.g.: "Header Extension Type" -> "headerExtensionType" (not "HE")
Objects of ASP type	Shall be derived from the name of the corresponding Service Primitive in the Standard, using any relevant abbreviations from the present annex. The full name as it appears in the core specification shall be included in parentheses after the name. E.g.: "CRLC-SUSPEND-Conf" -> "CRLC_SuspendCnf (CRLC-SUSPEND-Conf)" If the metatype PDU is not used, the ASP name shall reflect both the ASP, and the embedded PDU name, using an underscore to separate the ASP part from the PDU part. E.g. DataReq_StartDTMF_Ack for an RRC-DATA-Req with an embedded START DTMF ACKNOWLEDGE PDU
Objects of PDU type	Shall have exactly the same name as the Message it corresponds to in the standard. If this Message is named by more words, they shall be joined, leaving the blanks out E.g.: "AMD PDU" -> "AMDPDU"

E.2.3 Identifiers shall be distinguishing (use of prefixes)

To support rules 2, 3(b), 4, and 5, the prefixes shown in table E2 shall be used for TTCN objects. Prefixes are separated from the name by an underscore to improve readability by clearly separating the prefix from the name. This convention will also support searching operations. For example, a search for all uses of PIXIT parameters in the test suite is possible by searching for 'px_'.

The optional <*protocol*> part shall be included in the name when the object is closely related to the protocol (e.g. PICS, some PIXIT parameters), it is necessary to be unambiguous or improves comprehension significantly (e.g. no need to think about protocol stacks on all used interfaces during reading). The optional <*protocol*> part shall be used for types defined in common modules.

Table E2: Prefixes used for TTCN objects

TTCN object	Case of first character	Prefix	Comment
Test Suite	Upper	-	
TTCN Module	Upper	-	
Simple Type	Upper	[<protocol>_]	Note 8
Structured Type	Upper	[<protocol>_]	Note 8
Element in Structured Type	Lower	-	
ASN.1 Type	Upper	[<protocol>_]	Note 8
Element in ASN.1 Type	Lower	-	
Test Suite Operation	Upper	o_[<protocol>_]	Note 1, 8
TSO Procedural Definition	Upper	o_[<protocol>_]	Note 1, 8
Formal Parameter to TSO or TSOP	Upper	p_	
Test Suite Parameter (PICS)	Upper	pc_[<protocol>_]	Note 8
Test Suite Parameter (PIXIT)	Upper	px_[<protocol>_]	Note 8
Test Case Selection Expression	Upper	[<protocol>_]	Note 8
Test Suite Constant	Upper	tsc_[<protocol>_]	Note 8
Test Suite Variable	Upper	tsv_[<protocol>_]	Note 8
Test Case Variable	Upper	tcv_[<protocol>_]	Note 8
PCO Type	Upper	-	
PCO	Upper	-	Note 2
CP	Upper	cp_	Note 2
Timer	Upper	t_[<protocol>_]	Note 8
Test Component	Upper	mtc_[<protocol>_] or ptc_[<protocol>_]	Note 3, 8
Test Component Configuration	Upper	-	
ASP Type	Upper	[<protocol>_]	Note 4, 8
Parameters within ASP Type	Lower	-	Note 4
PDU Type	Upper	[<protocol>_]	Note 4, 8
Fields within PDU Type	Lower	-	Note 4
Encoding Definition	Upper	enc_	
Encoding Variation	Upper	var_	
Invalid Field Encoding Variation	Upper	inv_	
CM Type	Upper	cm_	
Field within CM Type	Lower	-	
Alias	Upper	a_	
ASP constraint	Upper	ca[b|d][s|r|w]_[<protocol>_]	Note 5, 8
Constraint (other types)	Upper	c[b|d][s|r|w]_[<protocol>_]	Note 5, 8
Formal Parameter for a Constraint	Upper	p_	
Test Case Group	Upper	<protocol>/	Note 8
Test Step Group	Upper		
Test Case	Upper	tc_	Note 6
Test Step	Upper	(ts_|pr_|po_)<CN domain>_<protocol>_	Note 7, 8, 9
Local tree	Upper	lt_	
Defaults	Upper	<protocol>_	Note 8

NOTE 1: Coding rules are not specified for test suite operation procedural definitions at this stage. These rules will be defined when the need arises

NOTE 2: A prefix is not used for PCO declarations, but is used for CP declarations. This is because PCOs and CPs will only be used in send and receive statements, and PCOs will be used more frequently than CPs. Since a PCO name or a CP name will be used on most behaviour lines, PCO names should be as short as possible – E.g. 2 to 3 characters.

NOTE 3: The prefix is mtc if the component role is MTC, or ptc if the component role is PTC. If multiple PTCs are used, the rest of the identifier will clarify which PTC is being referred to. E.g. ptc_Cell1, ptc_Cell2.

NOTE 4: This applies for both tabular and ASN.1 definitions.

NOTE 5: Constraint prefixes are built up from the following regular expression. c[a][b|d][s|r|w].

- 'c' shall always be present to indicate that the object is a constraint.
- 'a' shall be present for ASP constraints to distinguish them from PDU constraints.
- 'b' shall be present if the constraint is ever used as a base constraint. (i.e. included in the derivation path of any other constraint).
- 'd' shall be present if the constraint is derived from another constraint.(i.e. has an entry in it's derivation path field)
- 'b' and 'd' cannot both be used in the same constraint, thereby limiting the derivation path to 1.
- For the purpose of the present note, the following definitions are required (Ref TR 101 666 clause 12.6.2)
  - The term 'field' is used to represent a structured type element, an ASP parameter, or a PDU field.
  - A 'bound field' is a field that either contains a SpecificValue, or is Omitted (-).
  - An 'unbound field' is a field that contains any of the following matching mechanisms: Complement, AnyValue (?), AnyOrOmit (*), ValueList, Range, SuperSet, SubSet, AnyOne (?), AnyOrNone (*), Permutation, Length, or IfPresent.
- 's' may optionally be present if the constraint is only used in send statements. 's' shall not be present if the constraint contains any unbound fields, or any fields chained to a constraint whose prefix includes 'w' or 'r'.
- 'r' may optionally be present if the constraint is only used in receive statements.
- 'w' may optionally be present to indicate that the constraint contains fields that are unbound. Before these constraints are used in SEND events, all unbound fields must either be bound by using a derived constraint, or explicitly assigned a value in the SEND event behaviour line.
- Either 'w' or 'r' shall be used if any fields in the constraint are unbound or are chained to a constraint whose prefix includes 'w' or 'r'.

NOTE 6: Test case names will correspond to the clause in the prose that specifies the test purpose. E.g. tc_7_2_23_2. An additional digit may be specified if more than one test case is used to achieve the test purpose. If an additional digit is required, this probably means that the test prose are not well defined.

NOTE 7: Test steps may optionally use the prefixes pr_ or po_ to indicate that the test step is a preamble or postamble respectively.

NOTE 8: Protocol abbreviations are provided in table **Error! Reference source not found.**. Protocol abbreviations may optionally be used to clarify the scope of TTCN objects, or to resolve conflicts when the same name is required by multiple protocols within the ATS.
If two different types exist in the ATS that represent the same information (e.g. IMSI) conversion operations shall be used to ensure consistency between the types. Also, conversion operations shall be used to avoid asking the same PIXIT question twice. For example, if a type is defined as an OCTETSTRING[4] for a NAS protocol, and the same type is represented as a BITSTRING[32] for RRC, a single PIXIT question shall be asked, and conversion operations shall be used to ensure that the same value is used for both types.

Table E3: Protocol abbreviations for prefixes

Protocol / prefix
BMC
CC
CS
GMM
MAC
MM
PDCP
RLC
RRC
SMS
SS
TC

NOTE 9: The prefixes CS and PS may optionally be used to indicate that a test step is specific to circuit switched, or packet switched signalling respectively.

E.2.4 Identifiers should not be too long (use standard abbreviations)

To assist in keeping TTCN identifiers shorter, table E4**Error! Bookmark not defined.** provides a non-exhaustive set of standard abbreviations that shall be used when naming objects that are used in the body of dynamic behaviour tables. Consistent use of abbreviations will improve test suite readability, and assist maintenance.

Table E4: Standard abbreviations

Abbreviations	Meaning
acs	access
acp	accept
ack	acknowledge
act	activation
addr	address
(re)alloc	(re)allocated, (re)allocation
arg	argument
ass	assignment
auth	authentication
ava	avail, available
bCap	bearer capability
cau	cause
clg	calling
ch	channel
chk	check
ciph	cipher, ciphering
cld	called
clsmk	classmark
cmd	command
cmpl	complete
cnf	confirm
cfg	configuration
conn	connect
ctrl	control
def	default
descr	description
disc	disconnect
enq	enquiry
err	error
(re)est	(re)establish
ext	extended
fail	failure
ho	handover
id	identity / identification
ie	information element
iel	information element length
ind	indication
info	information
init	initialize
lvl	level
loc	location
locUpd	location update
max	maximum
mgmt	management
min	minimum
misc	miscellaneous
mod	modification
ms	mobile station
msg	message
mt	mobile terminal
neigh	neighbour
ntw	network
num	number

Abbreviations	Meaning
orig	origin/-al
pag	page/-ing
params	parameters
perm	permission
phy	physical
qual	quality
rand	random
ref	reference
reg	register
rej	reject
rel	release
req	request
rsp	response
rx	receiver
sel	selection
seq	sequence
serv	service
st	state
sysInfo	system information
sync	synchronization
sys	system
tx	transmitter

RULE 4: Specific naming rules for test suite parameters/constants/variables test case variables and formal parameters

a) The name should reflect the purpose/objective the object is used for.
b) If the type is not a predefined one, it is useful that the name reflects the type, too.
c) It could be useful, that the individual naming conventions are not the same for all object classes this rule applies to. e.g. use upper case letters for test suite parameters/constants, and use one of the other possibilities presented in ETR 141 example 1 for other object classes.

See also ETR 141 clauses 5.1-5.4 for further discussion on naming test suite parameters.

RULE 5: Specific naming rule for timers

If the timer is not defined in the protocol to be tested, the name should reflect the objective of the timer used for testing.
NOTE: There is no need to indicate the object type "timer" in the name, since timers only occur together with timer operations

RULE 6: Specific naming rule for PDU/ASP/structured types

As far as applicable, derivation rules or mapping tables should be used to relate the names of the types to the corresponding objects in the protocol or service definition.
NOTE: There may be types, e.g. erroneous PDU types, that do not relate to an object in the protocol or service definition.

Whenever names of types are derived from ASN.1 type definitions provided in the core specifications, the names shall remain the same as the ASN.1 specifications, and references shall be provided in the comment fields.

RULE 7: Specific naming rule for PDU/ASP/structured types constraints

Rules should be stated to derive the names from the names of the corresponding type definitions. It is often possible to use the type name plus an appropriate suffix reflecting the specific constraint value. In case of lengthy names, useful abbreviations or a defined numbering scheme can be chosen.

Constraint names begin with the appropriate prefix, followed by the first letter of each word in the type, followed by words describing the peculiarity of the constraint. E.g. Type = RadioBearerSetupPDU, constraint name could be cb_RBSP_GenericUM_DTCH.

RULE 8: Specific naming rule for test suite operations

The name should reflect the operation being performed.
i.e. the name should indicate an activity, not a status. This can be achieved e.g. by using appropriate prefixes like "check", "verify", etc.

RULE 9: Specific naming rule for aliases
The name should reflect that aspect of its expansion, that is important in the situation where the alias is used. Derivation rules should be provided to derive the alias name from its macro expansion or from the name of an embedded ASP / PDU.

See also ETR 141 clauses 6.3.6 and 9 for further guidelines on naming aliases.

RULE 10: Specific naming rule for test steps
The name should reflect the objective of the test step.

RULE 11: Selecting the ASN.1 format for type definitions
a) If the protocol standard uses ASN.1 to specify the PDUs, the ATS specifier should also use ASN.1. b) If the protocol standard does not use ASN.1, check carefully whether features of ASN.1 that the tabular format of type definition does not present are necessary in the ATS, or could ease the design and understanding of the definitions as a whole. Check especially whether fields or parameters have to be specified, the order of appearance of which, in a received ASP/PDU, cannot be predicted. If any of these conditions apply, use ASN.1 for type and ASP/PDU type declarations. c) Use the option of "ASN.1 ASP/PDU type Definitions by Reference" whenever applicable. d) Example 14 shows a compatibility problem that could occur, when ASN.1 type declarations as well as tabular type declarations are used in an ATS. Use the ATS Conventions to describe how this compatibility problem is handled in the ATS, i.e. whether in expressions and assignments entities defined in ASN.1 are only related to entities defined in ASN.1 or not.

Names of ASN.1 objects shall be kept the same as the core specifications in this case, even where the names are at odds with the naming conventions adopted for other TTCN objects.

RULE 12: Further guidelines on type definitions
a) Use simple type or ASN.1 type definitions whenever an object of a base type with given characteristics (length, range, etc.) will be referenced more often than once. b) Use the optional length indication in the field type or parameter type column of structured type and ASP/PDU type definitions whenever the base standard/profile restricts the length. NOTE 1: This can often be achieved by references to simple types. c) Map the applicable ASPs/PDUs from the service/protocol standard to corresponding ASP/PDU type definitions in the ATS. NOTE 2: It may happen that not all ASPs/PDUs of a service/protocol standard are applicable to a particular ATS for the related protocol. It may also happen that additional ASP/PDU type declarations are necessary, e.g. to create syntactical errors. d) Map the structure of ASPs/PDUs in the service/protocol standard to a corresponding structure in the ATS. NOTE 3: This mapping is not always one-to-one, e.g. because a field in the PDU definition of the protocol standard is always absent under the specific conditions of an ATS. But it should normally not happen, that a structured element in the protocol standard is expanded using the "<-" macro expansion, so that the individual fields are still referenced, but the structure is lost in the ATS.

RULE 13: Specification of test suite operations
a) Use a test suite operation only if it cannot be substituted by other TTCN constructs. b) Write down the rationale/objective of the test suite operation. Reference standards if applicable. c) Classify and simplify algorithm. Split test suite operation if too complex. d) Choose an appropriate specification language depending on the rationale/objective: - predicates for Boolean tests; - abstract data types for manipulation of ASN.1 objects; - programming languages for simple calculation. e) Check/proof the test suite operation: - is the notation used known/explained; - are all alternative paths fully specified; - is the test suite operation returning a value in all circumstances; - are error situations covered (empty input variables, etc.). f) State some evident examples.

E.2.5 Test suite operations must not use global data

All information required by test suite operations must be passed as formal parameters. This includes test suite variables, test case variables, test suite parameters, and constraints.

RULE 14: General aspects of specifying constraints
a) Develop a design concept for the complete constraints part, particularly with respect to the "conflicting" features as indicated in items i) to iv) and including naming conventions (see ETR 141 Clause 6).
b) Make extensive use of the different optional "Comment" fields in the constraint declaration tables to highlight the peculiarity of each constraint.

RULE 15: Relation between base constraints and modified constraints
a) Define different base constraints for the send- and receive direction of a PDU (when applicable).
b) Use modified constraints preferably when only a small number of fields or parameter values are altered with respect to a given base. NOTE 1: For SEND events the creation of a further modified constraint can sometimes be avoided, if an assignment is made in the SEND statement line, thus overwriting a particular constraint value.
c) Design the relation between base constraints and modified constraints always in connection with parameterization of constraints (see the two subsequent subclauses). NOTE 2: Additional parameters in a constraint, introduced to avoid the declaration of further base/modified constraints can reduce the amount of constraints needed in an ATS, but then the constraint reference is getting more and more unreadable.
d) When modified constraints are used, keep the length of the derivation path small. The length of the derivation path (resulting from the number of dots in it) is a kind of nesting level, and it is known from experience that a length greater than 2 is normally difficult to overview and maintain.

Modified constraints should not have a derivation path longer than 1. A modified constraint should not alter more than 5 values with respect to a given base constraint. If a constraint is used as a base constraint, it must have the prefix 'cb', to warn test suite maintainers / developers that any changes to this constraint may cause side effects.

Note that if an existing constraint without the 'cb' prefix is to be used as a base constraint, either a new, identical constraint with an 'cb' prefix must be created, or the existing constraint must be renamed to include the 'cb' prefix in all places it is referenced in the test suite.

[Outstanding issue: Stating that all base constraints must have the 'cb' prefix may result in ALL constraints having this prefix, in case they are used as base constraints at a later stage during devlopment. If this occurs, the 'b' becomes redundant, and should be removed from the coding conventions.]

RULE 16: Static and dynamic chaining
a) Make a careful evaluation of which embedded PDUs are needed in ASPs/PDUs, in which (profile) environment the ATS may operate and which kind of parameterization for other parameters/fields is needed, to find an appropriate balance between the use of static and/or dynamic chaining in a particular ATS.
b) When the ATS is used in different profile environments and the types and values of embedded PDUs cannot be predicted, dynamic chaining is normally the better choice.
c) When static chaining is used, chose the name of the ASP/PDU constraint such that it reflects the peculiar value of the embedded PDU (see also the clause on naming conventions in ETR 141).

RULE 17: Parameterization of constraints
a) Make a careful overall evaluation of which field/parameter values are needed in ASPs and PDUs to find an appropriate balance between the aim of a comparably small number of constraint declarations and readable and understandable constraint references.
b) Keep the number of formal parameters small. Keep in mind, that the number of formal parameters in structured/ASN.1 types Constraints will add up to the total number of ASP/PDU constraints. A clear border for the number of formal parameters cannot be stated, but it is known from experience that a number bigger than 5 normally cannot be handled very well.

Constraints should not be passed more than five parameters. Instead, more constraints should be defined. Related parameters can be grouped in new structured types to reduce the number of parameters that must be passed to constraints.

[NOTE: The value five has been selected based on the recommendation in ETR 141 rule 17. If more parameters are required, we can update this rule, or use more than 5 parameters, and provide documentation indicating why more parameters are required.]

A constraint should not be passed parameters to that are not processed in that constraint. If for example a parameter is to be passed from a PDU constraint to a structured type constraint then the PDU constraint should be made specific and not have that parameter passed. The reason for this is that no editors as yet can trace through this mechanism and it becomes very difficult in a complex suite to see exactly what is being passed.

For example:

```
PduA ::= SEQUENCE {
  infoElement1  InformationElementType1,
  infoElement2  INTEGER
}

InformationElementType1 ::= SEQUENCE {
  field1  INTEGER,
  field2  INTEGER
}

cb_PATypical( p_Field1: INTEGER; p_Field2: INTEGER ) ::= {
  infoElement1  c_IET1Typical( p_Field1 ),
  infoElement2  pField2
}

c_IET1Typical( p_Field1: INTEGER ) ::= {
  field1  p_Field1,
  field2  5
}
```

In the example constraint cb_PATypical, passing p_Field1 through to a nested constraint is not allowed, but the use of p_Field2 is acceptable.

RULE 18: Constraint values
a) Use comments to highlight the peculiarity of the value, especially when the value is a literal, whose meaning is not apparent. b) Use test suite constants instead of literals, when appropriate. Normally not all literals can be defined as Test Suite Constants, but a rule by thumb is: if a literal value of a given type occurs more than once (as a constraint value or more generally in an expression), then it is useful to define it as a Test Suite Constant, letting the name reflect the value. c) Use the length attribute when possible and when the length is not implicit in the value itself or given by the type definition (e.g. for strings containing "*").

RULE 19: Verdict assignment in relation to the test body
Make sure that verdict assignment within a default tree is in relation to the test body. If an unsuccessful event arising in the test body is handled by the default tree, then assign a preliminary result "(FAIL)" within the corresponding behaviour line of the default tree. If the position of the unsuccessful event is not in the test body, assign a preliminary result "(INCONCLUSIVE)". If the behaviour line handling the unsuccessful event is a leaf of the default tree, assign a final verdict instead.

RULE 20: Test body entry marker
The entry of the test body should be marked.

RULE 21: State variable
For realizing test purposes dependent on protocol states, use a variable to reflect the current state of the IUT.

RULE 22: State checking event sequences
Combine event sequences used for checking a state of the IUT within test steps.

RULE 23: Easy adaptation of test steps to test cases
For easy adaptation of a test step to test case needs, parameterize the constraints used within a test step.

Test steps may be parameterised, but with no more than five parameters. See also ETR 141 clause12.2 and rule 28. Related parameters can be grouped in new structured types to reduce the number of parameters that must be passed to constraints.

[NOTE: Again, the value five has been selected based on the recommendation in ETR 141 rule 17. If more parameters are required, we can update this rule, or use more than 5 parameters, and provide documentation indicating why more parameters are required.]

RULE 24: Minimizing complexity of test steps
Minimize the complexity of test steps either by restricting the objective of a test step to atomic confirmed service primitives or by separating event sequences which build different "logical" units into different test steps.

RULE 25: Nesting level of test steps
Keep the nesting level of test steps to a minimum.

RULE 26: Recursive tree attachment
Avoid recursive tree attachment. Where possible, use loops instead of recursive tree attachments.

RULE 27: Verdict assignment within test steps
If verdicts are assigned within a test step, guarantee at least the partial (i.e. not general) re-use of the test step.

RULE 28: Parameterized test steps
Use parameterized test steps to ensure re-use of test steps within test cases for different needs.

RULE 29: Combining statements in a sequence of alternatives
If there is no Boolean expression included in an alternative sequence, a statement of type UCS should never be followed by a statement of type UCS or CS within a sequence of alternatives.

RULE 30: Using relational expressions as alternatives
a) A relational expression should never restrict the value range of a preceding relational expression in the same alternative sequence using the same variable. b) The value range of a relational expression should be different from the whole value range of all preceding relational expressions in the same alternative sequence using the same variable.

RULE 31: Loop termination
Do not use conditions for terminating loops, which depend only on the behaviour of the IUT.

RULE 32: Avoiding deadlocks
a) Make sure that each alternative sequence of receive events contains an OTHERWISE statement (without any qualifier) for each PCO. b) Make sure that each alternative sequence of receive events contains at least one TIMEOUT event (implying that a corresponding timer was started).

A set of alternatives using qualifiers shall always include an alternative containing the qualifier [TRUE], to provide a default behaviour if none of the qualifiers match.

For example:

```
[ tcv_Value = 1 ]
  AM ! ASP_ForValue1
    ...
[ tcv_Value = 2 ]
  AM ! ASP_ForValue2
    ...
[ TRUE ]
  AM ! ASP_ForOtherValues
    ...
```

RULE 33: Straightforward specification of test cases
a) Use only event sequences leading to the test body within a preamble. b) Handle all event sequences not leading to the test body within the default tree of the test case/step. c) If the very same event sequence can be used to transfer the IUT from each possible state to the idle state, then realize this event sequence as a postamble.

RULE 34: Test component configuration declaration
Avoid recursive test component configuration declarations.

RULE 35: Default trees with RETURN statement
Special care should be taken by using a RETURN statement within a default tree in order to avoid an endless loop resulting from the expansion of the default tree.

E.3 3GPP ATS implementation guidelines

This clause provides a set of guidelines that must be followed during ATS development. In general, these guidelines are intended to prevent developers from making common errors, or discuss considerations that must be taken into account before using specific features of the TTCN language.

E.3.1 Test case groups shall reflect the TSS&TP document

Test groups shall be used to organise the test cases in the same way as the test purposes are structured in the prose specification.

The general structure of the test groups should be in the following format.

<protocol>/<group>/<subgroup>

E.g. RLC/UM/Segmentation/LengthIndicator7bit/

E.3.2 Test case names correspond to the clause number in the prose

Test case names are derived directly from the clause number in the prose specification. Decimal points between digits in the clause number are replaced with underscores. E.g. the test case name for the test purpose specified in clause 7.2.3.2 of 34.123-1 is tc_7_2_3_2. If more than one test case is required to achieve a test purpose, an additional digit may be added. See also ETR 141 clause 6.3.7

E.3.3 Use standard template for test case and test step header

Table E5 illustrates how the Test Case dynamic behaviour header fields should be used.

Table E5 – Template for TTCN test case table header

Field			Contents			
Test Case Name:			tc_NUMBER_OF_TESTCASE The number of the test case, which is used in the name of the test case, is the number it has in the prose specification. e.g.: "tc_26_13_1_3_1"			
Group:			Is automatically filled and cannot be changed			
Purpose:			This is taken directly from the prose specifications.			
Configuration:			As required if concurrent TTCN is being used.			
Default			The appropriate default			
Comments:			First line contains: Specification: The names and sub clauses of relevant core specifications. Next line contains: Status: OK / NOT OK (+explanation if not ok) / Version number / Validated / Reviewed etc… E.g.: Status: OK Rest of lines give comments as: What has to be done before running this test? E.g.: 1. Generic setup procedure must be completed before running this test. Any special information about what might be needed for the testing system, like specific requirements for the testing system, specific hacks, certain settings etc. This field should be short (if long description is needed it must be put into Detailed Comments)			
Selection Ref:			The appropriate test case selection expression.			
Description:			Optional. Max 4 lines. If available, this should be the title of the prose clause. Note 11			
Nr	Label	Behaviour Description		Constraints Ref	Verdict	Comments
1		Note 13		Note 13		Note 11
Detailed Comments			Contains detailed information about test steps + additional information Note 11			

NOTE 11: The description field in the test case / step header is used to generate the test suite overview, and should only include a brief overview of the test case / step with a maximum of 4 lines. For a more detailed description of the test case / step algorithm / parameters etc, the comments or detailed comments fields should be used.

NOTE 11: The comments field for each behaviour line should usually consist of a number that is a reference to a specific numbered comment in the detailed comments field. If this extra level of indirection reduces readability, brief comments can be used in the comments field for each behaviour line.

NOTE 13: If entries in the behaviour description or constraints reference column contain lists with more than one element, carriage returns should be used between list elements to prevent the line from becoming too long.

Table E6 illustrates how the Test Case dynamic behaviour header fields should be used.

Table E6: Template for TTCN test case table header

Test Step Name		ts_TestStepName(p_Param1: Param1Type; p_Param2: Param2Type)			
Group		Is automatically filled and cannot be changed			
Objective		The objective of the test case. Provides a brief summary of the functionality of the test step.			
Default		The appropriate default			
Comments		A detailed description of the test step, including the relevant items from the following categories: Algorithm A detailed description of the algorithm / principles used within the test step Parameters: A description of each of the parameters passed to the test step, including the purpose of the parameter, valid values, restrictions etc. Preconditions The required state of the UE and / or SS before using this test step, including test steps that should be executed before using the present test step, and a description of all test case variables that must contain appropriate values before using this test step. Postcondidions The expected state of the UE and / or SS after using this test step, including a description of all test case variables that will be modified by this test step. NOTE: It is too difficult to maintain the list of variables required / affected by nested test steps, so it is the users responsibility to check which variables are required / affected by nested test steps.			
Description		Optional. Max 4 lines. Note 11			
Nr	Label	Behaviour Description	Constraints Ref	Verdict	Comments
1		Note 13	Note 13		Note 11
Detailed Comments		Contains detailed information about test steps + additional information Note 11			

E.3.4 Do not use identical tags in nested CHOICE constructions

A nested CHOICE requires tags in the different alternative type lists to differ (see ISO 8824, sub clause 24.4, EXAMPLE 3, INCORRECT). 'The tag shall be considered to be variable, ... becomes equal to the tag of the "Type" ... from which the value was taken.'

EXAMPLE: components are defined in a nested CHOICE construction, but no distinguishing tags are used to make the difference between component types, i.e. tags for different types turn out to be identical.

```
Component ::=  CHOICE {
  gSMLocationRegistration_Components   GSMLocationRegistration_Components,
  gSMLocationCancellation_Components   GSMLoactionCancellation_Components,
  ...
}

GSMLocationRegistration_Components ::= CHOICE {
  gSMLocationRegistration_InvokeCpt    [1] IMPLICIT GSMLocationRegistration_InvokeCpt,
  gSMLocationRegistration_RRCpt        [2] IMPLICIT GSMLocationRegistration_RRCpt,
  gSMLocationRegistration_RECpt        [3] IMPLICIT GSMLocationRegistration_RECpt,
  gSMLocationRegistration_RejectCpt    [4] IMPLICIT RejectComponent
}

GSMLocationCancellation_Components ::= CHOICE  {
  gSMLocationCancellation_InvokeCpt  [1] IMPLICIT GSMLocationCancellation_InvokeCpt,
  gSMLocationCancellation_RejectCpt  [4] IMPLICIT RejectComponent
}
```

gSMLocationRegistrationInvokeCpt and gSMLocationCancellation_InvokeCpt have the same tag and can therefore not distinguished anymore. Note that ITEX 3.5 does not report this error.

E.3.5 Incorrect usage of enumerations

Enumerations may contain distinct integers only (see ISO 8824, sub clause 15.1)

EXAMPLE: TypeOfNumber containing a NamedValueList in which there are non-distinct values.

```
TypeOfNumber ::=  ENUMERATED {
.....,
  internationalnumber  (1),
  level2RegionalNumber (1),
  nationalNumber       (2),
  level1RegionalNumber (2),
......
 }
```

E.3.6 Structured type as OCTETSTRING should not be used

"It is required to declare all fields of the PDUs that are defined in the relevant protocol standard, …"
TR 101 101 TTCN specification clause 11.15.1

EXAMPLE: The ISDN Bearer Capability Information Element (BCAP) contents is defined as OCTETSTRING.

EXAMPLE: Usage of data type BITSTRING [7..15] as data type of the Call Reference (= 7 bits or =15 bits, but not 8 bits for example) does not correspond to the specification !!).

E.3.7 Wildcards in PDU constraints for structured types should not be used

Values ? and * are incorrect in PDU constraints where they are used to indicate values of structured types, because they would allow any combinations of values – even incorrect ones - which is not admissible according to the specifications. It is to be kept in mind that in tabular form each field is optional! It would be better to create and use an "any"-constraint which would deal with all the fields in detail (mandatory, IF PRESENT, etc.). See TBR 33, 34 as example.

[TODO: Use of wildcards vs. 'any' constraints requires some further discussion]

E.3.8 TSOs should be passed as many parameters as meaningful to facilitate their implementation

Parameters should be passed to TSOs to facilitate the TSO realization. If a TSO is used in various contexts, this should be reflected in the parameters passed to the TSO. Specifically, TSOs operating on well-defined (parameterized) constraints should take these constraints (including relevant parameters) as parameters if required.

BAD EXAMPLE: In this example, the TSO may be used in many contexts, but no information is passed to the TSO, which makes TSO realization difficult.

		L?SETUPr (... tcv_invokeId := TSO_GET_INVOKEID ( ), ...)	Sr (SU_GR3(GSM_IncomingCallMMInfo_Invoke(...)))		

GOOD EXAMPLE: In this case, the TSO is provided with information about the data object from which the invoke Id is to be extracted, and the type of component from which the invoke Id is to be extracted is identified by passing the component constraint.

		L?SETUPr (... tcv_invokeId := TSO_GET_INVOKEID (DL_DataInd_Setup.msg, GSM_IncomingCallMMInfo_Invoke(...)), ...)	Sr (SU_GR3(GSM_IncomingCallMMInfo_Invoke(...)))		

To calculate the invocation identification and store the result in variable tcv_invokeId the TSO has to be provided with

information about the data object from which the invoke Id is to be extracted. PDU constraint SU_GR3 may contain several components. In the specific situation only one of these components is relevant.

Depending on the nature of the TSO, passing the received value, or a subcomponent of the received value may be more appropriate than passing the constraint.

E.3.9 Specification of Encoding rules and variation should be indicated

TTCN does not mandate encoding rules, although TTCN foresees that applicable encoding rules and encoding variations can be indicated for the data structures used in a test suite.

There are standards defining encoding rules, e.g. the ITU-T X.680 series. However, the type of encoding called "Direct Encoding" - a bit-by-bit-mapping from the data definitions onto the data stream to be transmitted - is not defined anywhere. It therefore needs a "home".

TTCN should therefore define which encoding rules may legally be used by TTCN test suite specifiers. All the encoding rules defined in the X.680 series should be contained in this repertoire. Additionally an encoding rule called Direct Encoding is needed in particular for tabular TTCN.

X.680 allows to encode data objects using different length forms (short, long, indefinite). These could be used alternatively as encoding variations. Another encoding variation could be the "minimum encoding", accepting any of the length forms in reception, and using the shortest of the available forms in sending. The variation actually used has to be described somewhere (in the ATS).

EXAMPLE: In EN 301 144-4 BER is used as encoding variation, but is in fact an encoding rule. Furthermore no default encoding is specified which would apply to those data structures for which no encoding rule or encoding variation is explicitly specified.

E.3.10 Use of global data should be limited

The Phase 2 ATS became extremely complex due to the global definition of data. Data should be defined locally where possible if the language allows, alternatively the names of global constraints could be given prefixes to indicate their use.

E.3.11 Limit ATS scope to a single layer / sub-layer

Separate ATSs should be produced to test each Layer and perhaps sub Layer. By doing this preambles and common areas particular to one sub Layer can be confined to one test suite and parallel development of test suites can be facilitated.

E.3.12 Place system information in specially designed data structures

System Information data could be stored in specially defined data structures, use of these structures to build PDUs may help to ensure that a consistent set of data is transmitted in all the channels in a cell.

E.3.13 Place channel configuration in specially designed data structures

Likewise the configuration of a 'channel' could be stored in similar structures. This data can then be used to configure the test system and to build Assignment messages to the UE under test. This may help avoid the situation where the TTCN creates one channel and unintentionally commands the mobile to a different, non-existent, channel.

E.3.14 PICS / PIXIT parameters

It is desirable to limit the scope of PICS / PIXIT parameters.

A default value shall be provided in the PIXIT document for all PIXIT parameters.

PICS / PIXIT parameters shall not include structured types. If a structured parameter is required, several parameters shall be used, one for each simple element within the type, and a constraint shall be created to combine the simple parameters into a structured type.

For example, to use the following structured type as a parameter:

Type Name	LocAreaId_v		
Encoding Variation			
Comments	Location Area Identification Value 3GPP TS 24.008 cl. 10.5.1.3		
Element Name	**Type Definition**	**Field Encoding**	**Comments**
mcc	HEXSTRING[3]		MCC 3 digits
mnc	HEXSTRING[3]		MNC 3 digits
lac	OCTETSTRING[2]		LAC
Detailed Comments			

The following three PIXIT parameters should be defined: Parameter Name	**Type**	**PICS/PIXIT Ref**	**Comments**
px_LACDef	OCTETSTRING	PIXIT TC	default LAC
px_MCCDef	HEXSTRING	PIXIT TC	default MCC
px_MNCDef	HEXSTRING	PIXIT TC	default MNC

And then the following constraint can be used to combine the simple parameters into a structured parameter:

Constraint Name	cb_LocAreaIdDef_v		
Structured Type	LocAreaId_v		
Derivation Path			
Encoding Variation			
Comments			
Element Name	**Element Value**	**Element Encoding**	**Comments**
mcc	px_MCCDef		
mnc	px_MNCDef		
lac	px_LACDef		
Detailed Comments			

E.3.15 Dynamic vs. static choices

Don't use wildcards for static choice constraints. For example, a type that is similar for FDD and TDD should have 2 type definitions, rather than a single type that uses an ASN.1 choice. Then in the TTCN, the correct type should be selected based on test suite parameters.

E.g.:

```
[ pxUseTddMode ] AM ! TddSpecificAsp
  AM ?
    ...
[ pxUseFddMode ] AM ! FddSpecificAsp
  AM ? …
    ...
```

E.3.16 Definition of Pre-Ambles and Post Ambles

Test cases should, as far as possible, use one of a set of standard pre-ambles to place the user equipment in its initial conditions. These pre-ambles should align with the generic setup procedures in the conformance specification. All non-standard pre-ambles should be identified and added to the pre-amble library.

With pre-ambles readability is very important so they should not use other test steps to send message sequences, and they should be passed as few parameters as possible. This also makes the results log easier to read.

The prose message sequence charts should be analysed, and a catalogue of common ways in which the test cases can terminate (correctly or incorrectly) created. This catalogue should be used to create a set of post-ambles. All final verdicts should be assigned in the post-ambles.

Wherever possible, a post-amble should return the test system and the User Equipment under test to a known idle state.

E.3.17 Use test steps to encapsulate AT and MMI commands

When the same AT or MMI command is to be used more than once within a test suite, the command should be placed within a test step, to ensure that the same information is provided consistently. The main intention of this guideline is to ensure that MMI commands provided to the user are consistent, and can be changed easily if required.

For example, a test step similar to the one illustrated in table E7 should be created and attached so that the same information is provided to the user each time the test step is used, and the string to be sent only exists in one place within the test suite.

Table E7: Example test step to encapsulate AT / MMI commandsDefault behaviour

Test Step Name		ts_AT_MMI_Example			
Group					
Objective		Send an MMI command instructing the user to insert the USIM card into the UE.			
Default					
Comments		Encapsulate an AT / MMI command within a test step to ensure that the same information is used consistently, and the information only exists in one place within the test suite.			
Description					
Nr	**Label**	**Behaviour Description**	**Constraints Ref**	**Verdict**	**Comments**
1		Ut ! MMI_CmdReq	ca_MMICmdReq (" Please insert the USIM card into the UE ")		
2		Ut ? MMI_CmdCnf	ca_MMICmdCnf		

Defaults are test steps that are executed when ever a receive event occurs that is not expected. Not expected means that it does not match any of the defined ASP constraints at that point in the test case. The default behaviour used in test case is defined in the test case declaration. They can be defined to stop the test case by calling a standard post-amble or receive the event as OTHERWISE and RETURN back to step where the unexpected event occurred.

A strategy for dealing with unexpected behaviour involving consistent use of defaults should be developed, and applied to test cases wherever possible.

If during a test case or test step it is necessary to change the default behaviour, the ACTIVATE statement may be used.

E.3.18 Use system failure guard timers

A timer should be set at the beginning of each test case to guard against system failure. Behaviour on expiry of this timer should be consistent for all test cases.

E.3.19 Mapping between prose specification and individual test cases

The ATS should map one-to-one between test cases and tests as described in 32.123-1. A method for ensuring that the two specifications track each other needs to be defined.

E.3.20 Verdict assignment

E.3.20.1 General

Final verdicts shall only be used to indicate test case errors, or when unexpected UE behaviour occurs such that it not sensible to continue the test. When a test case reaches a leaf node, the test case ends, and the current preliminary verdict is assigned. At least one preliminary verdict shall be assigned for every test case. If a test case terminates and no final or preliminary verdicts have been assigned, the current value of the predefined variable R will be 'none', and a test case error is recorded instead of a final verdict.

Labels shall be used for every line in which a verdict is posted to improve the traceability of the conformance log produced when the test case is executed. These labels should be kept short, since they appear in the dynamic behaviour tables.

All test suites shall make use of a global boolean variable, defined in the common module, called tcv_TestBody. tcv_TestBody is updated within each test case to indicate if the test body is currently being executed. tcv_TestBody is referenced in defaults and test steps to assign a preliminary inconclusive verdict when unexpected events occur outside of the test body, or a preliminary failure verdict when unexpected events occur within the test body.

The initial value in the declaration of the test case variable tcv_TestBody shall be FALSE. The variable will be bound to this value when the ATS is initialised, and will be re-bound to this value after termination of each test case, ready for execution of the next test case.

E.3.20.2 Test cases

A line similar to line 3 in table E8 shall be used in all test cases to set tcv_TestBody to TRUE. This line shall have the label TBS to indicate the Test Body Start point.

A line similar to line 6 in table E8 shall be used in all test cases to set tcv_TestBody to FALSE. This line shall have the label TBE[N] to indicate the Test Body End point. A number N (with one or more digits) may optionally be appended to the label to distinguish between multiple test body end points. If the number of possible test sequences makes management of the tcv_TestBody variable too difficult, the variable can be set to TRUE at the beginning of the test. In this case, a comment shall be added to the test case noting that tcv_TestBody is not updated, so verdicts assigned within preambles and postambles will be treated as if they are part of the test body.

Within the test body, preliminary verdicts shall be used to indicate the result of the test purpose. Each behaviour line within the test body containing a preliminary verdict shall have a label of the form TBXN, where X is one of P, F, I for pass, fail, and inconclusive respectively, and N is a number (with one or more digits) used to distinguish multiple TBPs, TBFs, or TBIs in the same test case.

If an unexpected event occurs corresponding to a test case error, a final inconclusive verdict shall be assigned, and the behaviour line shall have a label ERRN, where N is a number used to distinguish multiple ERRs, and ERR indicates that a test case error has occurred. An example of this is provided in the test step clause.

Table E8 contains an example test case illustrating these concepts.

Table E8: Example test case illustrating use of verdicts, labels and tcv_TestBody test case variable.

Nr	Label	Behaviour Description	Constraints Ref	Verdict	Comments
1		+ts_Preambles			
2	TBS	(tcv_TestBody := TRUE)			1
3		L ! Stimulus	cs_Stimulus1		
4		+lt_Response			
5	TBE	(tcv_TestBody := FALSE)		(P)	2
6		+ts_Postambles			
		lt_Response			
7	TBP1	L ? Response	cr_ValidResponse1	(P)	3
8	TBP2	L ? Response	cr_ValidResponse2	(P)	3
9	TBF1	L ? Response	cr_InvalidResponse	(F)	4
10	TBI1	L ? Response	cr_OtherResponse	(I)	5
Detailed comments		1. The behaviour line setting tcv_TestBody to TRUE shall have the label TBS 2. The behaviour line setting tcv_TestBody to FALSE shall have the label TBE, and can optionally be used to assign a verdict indicating that the test purpose has passed or failed (i.e. if the final behaviour statement in the test body is a tree attachment) 3. The label TBPN is used to indicate that the test purpose has been achieved via the Nth possible valid UE behaviour 4. The label TBFN is used to indicate that the test purpose has not been achieved, due to the Nth possible failure cause 5. The label TBIN is used to indicate that the test result is inconclusive for the Nth possible unexpected / unknown event			

E.3.20.3 Test steps

To promote re-use, test steps shall only assign preliminary verdicts (I) and (F). (P) verdicts shall be managed at the test case level in general, but may be used sparingly within test steps. ETR 141 clause 12.4 recommends that a preliminary pass verdict should be assigned at the leaf of each passing event sequence of the test step. If a test step includes an alternative for unexpected / invalid behaviour, then either a preliminary inconclusive verdict shall be assigned if tcv_TestBody is FALSE, or a preliminary failure verdict shall be assigned if tcv_TestBody is TRUE.

Each behaviour line within the test step containing a preliminary verdict shall have a label of the form TSXN, where X is one of P, F or I for pass, fail, and inconclusive respectively, and N is a number (with one or more digits) used to distinguish multiple TSPs, TSFs, or TSIs in the same test step.

If an unexpected event occurs corresponding to a test case error, a final inconclusive verdict shall be assigned, and the behaviour line shall have a label ERRN, where N is a number used to distinguish multiple ERRs, and ERR indicates that a test case error has occurred.

Table E9 contains an example test step illustrating these concepts.

Table E9: Example test step illustrating use of verdicts, labels and tcv_TestBody test case variable.

Nr	Label	Behaviour Description	Constraints Ref	Verdict	Comments
1		[p_Mode = tsc_Mode1]			
2		L ! Stimulus	cs_Stimulus1		
3		+lt_Response			
4		[p_Mode = tsc_Mode2]			
5		L ! Stimulus	cs_Stimulus2		
6		+lt_Response			
7	ERR1	[TRUE]		I	1
		lt_Response			
8		L ? Response	cr_ValidResponse1		2
9		L ? Response	cr_InvalidResponse		
10	TSI1	[tcv_TestBody = FALSE]		(I)	3
11	TSF1	[tcv_TestBody = TRUE]		(F)	4

Detailed comments	
	1. An invalid value for the parameter p_Mode has been passed to this test step, so a final inconclusive verdict is assigned, with a label indicating that a test case error has occurred. 2. If the expected behaviour occurs, then the test step completes at the leaf node, and the current preliminary verdict is not changed. 3. If unexpected / invalid behaviour occurs, and the current test step is being used as a preamble or postamble (tcv_TestBody = FALSE) then a preliminary inconclusive verdict is assigned. 4. If unexpected / invalid behaviour occurs, and the current test step is being used as part of the test purpose(tcv_TestBody = TRUE) then a preliminary failure verdict is assigned.

E.3.20.4 Defaults

Each behaviour line within a default behaviour table containing a preliminary verdict shall have a label of the form DFXN, where X is one of F or I for fail, and inconclusive respectively, and N is a number (with one or more digits) used to distinguish multiple DFFs, or DFIs in the same test step.

tcv_TestBody shall be referenced from within default behaviour tables to assign the appropriate verdict when unexpected events occur.

Table E10 contains an example default behaviour table illustrating these concepts.

TableE10: Example default behaviour table illustrating use of verdicts, labels and tcv_TestBody test case variable

Nr	Label	Behaviour Description	Constraints Ref	Verdict	Comments
1		L ? Response	cr_IgnoredResponse		1
2		RETURN			
3	DFI1	L ? OTHERWISE [tcv_TestBody = FALSE]		(I)	2
4	DFF1	L ? OTHERWISE [tcv_TestBody = TRUE]		(F)	3

Detailed comments	
	1. Valid events that are to be ignored can be included in the default behaviour, but should have no preliminary verdict assigned. 2. If unexpected data is received in the preambles or postambles, a preliminary inconclusive verdict is assigned, and the test case is terminated 3. If unexpected data is received in the test body, a preliminary failure verdict is assigned, and the test case is terminated

[TODO: Update verdict assignment clause to include use of concurrent TTCN when required.]

See also ETR 141 clauses 11.2, 12.4, and 14.3.

E.3.21 Test suite and test case variables

A default value shall be provided for all test suite and test case variables.

[TODO: Define rules for use of test suite and test case variables, especially with regard to concurrent and modular TTCN]

E.3.22 Use of macros is forbidden

The use of macros is forbidden, to support migration to TTCN3.

E.3.23 Support for future Radio Access Technologies

To allow existing test cases to be updated in future to support other radio access technologies, test suites shall make use of a PIXIT parameter px_RAT of type RatType as shown in the following example.

Test Case Name		tc_RAT_Example1			
Nr	Label	Behaviour Description	Constraints Ref	Verdict	Comments
1		START t_Guard(300)			
2		[px_RAT = fdd]			
3		PCO ! FDD_PDU	c_FDD_PDU1		FDD specific behaviour
4	TBP1	PCO ? COMMON_PDU	c_COMMON_PDU1	(P)	
5		[px_RAT = tdd]			
6		PCO ! TDD_PDU	c_TDD_PDU1		TDD specific behaviour
7	TBP2	PCO ? COMMON_PDU	c_COMMON_PDU1	(P)	
8		[px_RAT = other_rat]		I	Tests for this RAT not implemented yet
9	TCE1	[TRUE]		I	Unexpected px_RAT value
Detailed Comments					

In general, alternatives should be used to separate behaviour specific for each RAT, and common behaviour should be re-used as much as possible. A final inconclusive verdict shall be used for any alternatives that have not been implemented yet.

Local trees may be used as shown in the following example to improve re-use of common behaviour.

Test Case Name		tc_RAT_Example2			
Nr	Label	Behaviour Description	Constraints Ref	Verdict	Comments
1		START t_Guard(300)			
2		+lt_RAT_SpecificPart			
3	TBP1	PCO ? COMMON_PDU	c_COMMON_PDU1	(P)	Common behaviour
		lt_RAT_SpecificPart			
4		[px_RAT = fdd]			
5		PCO ! FDD_PDU	c_FDD_PDU1		FDD specific behaviour
6		[px_RAT = tdd]			
7		PCO ! TDD_PDU	c_TDD_PDU1		TDD specific behaviour
8	TCE1	[TRUE]		(I)	Unexpected px_RAT value
Detailed Comments					

E.3.24 Managing multiple representations of the same information.

When the same information is represented using multiple types within the same test suite, it is necessary to manage conversions between the types, and ensure that the information remains consistent across all of the representations.

For example, IMSI is represented as 'SEQUENCE (SIZE (6..15)) OF Digit' in the RRC ASN.1 definitions, as a HEXSTRING for input as a PIXIT parameter, and as an information element defined in TTCN tabular format for MM.

E.3.24.1 Predefined types

Conversion operations are not required to convert the following TTCN predefined types to their counterparts in ASN.1.

a) INTEGER predefined type

b) BOOLEAN predefined type

c) BITSTRING predefined type

d) HEXSTRING predefined type

e) OCTETSTRING predefined type

f) OBJECTIDENTIFIER predefined type

g) R_TYPE predefined type

h) CharacterString predefined types

Therefore it is valid to pass a value of type BIT STRING (ASN.1) as a formal parameter of type BITSTRING (TTCN predefined).

E.3.24.2 Simple types

TR 101 666 clause 11.2.1 states:

> 'TTCN is a weakly typed language, in that values of any two types which have the same base type are considered to be type compatible (e.g. for the purposes of performing assignments or parameter passing).'

When simple types have restrictions, it is the TTCN author's responsibility to ensure that the restrictions are compatible. The TTCN compiler provides some assistance with this, but the extent of the checking is compiler specific.

E.3.24.3 Structured types

For conversion between more complex representations, test suite operations will generally be required. If the mapping is simple enough, it may be possible to perform the conversion using a test step, which takes the common representation as a parameter, and stores the required representation in a test case variable. This may avoid the need for an extra test suite operation.

E.3.24.4 Conversion responsibility

Two design approaches are possible for deciding where the responsibility of conversion lies: Calling party conversion and called party conversion.

The appropriate option should be selected on a case-by-case basis with the following restrictions:

- If one representation of the information is a PIXIT parameter, and this information must be passed to a test step, the called party conversion option shall be used, and the formal parameter to the test step shall always have the same type as the PIXIT parameter.
- If a test step provides multiple alternatives for different radio access technologies, which require different representations of the same information, the called party conversion convention shall be used. In this case a technology independent representation of the information shall be passed as a parameter, and the test step shall perform the conversion to the appropriate type depending on which RAT is being used.

E.3.24.5 Option 1: Calling party conversions

For this approach, each test step provides an interface based on its internal representation. It is the responsibility of the test case / step attaching the test step to perform the conversion before the attachment.

E.3.24.5.1 Advantages

- The number of calls to conversion operations is minimised.
- The complexity of the attached test steps is reduced because fewer conversions are required than for the called party conversion approach.

E.3.24.5.2 Disadvantages

- Different types are used to transfer the same information across the test step interfaces.
- The complexity of the attaching test steps / cases may be increased because conversions are required before attaching a test step.
- The attaching test steps / cases are responsible for ensuring that multiple representations contain consistent information.

E.3.24.6 Option 2: Called party conversions

In this case, the same representation is used wherever the information must be used as a formal parameter value to a test step, and it is the responsibility of the test step to perform any conversions required.

E.3.24.6.1 Advantages

- The complexity in the attaching test case / step is reduced, which will often improve readability.
- The test step interfaces are cleaner, because the same representation is always passed as a formal parameter.
- Internal representations may be hidden within test steps so that calling parties do not need to have any knowledge of them.

E.3.24.6.2 Disadvantages

- Conversion operations may be called more times than necessary, for example if the same test step is attached twice within one test case.

Annex J (informative): Change history

Change history					
TSG T#	Version	CR	Tdoc T	New Version	Subject/Comment
	-	-			