High performance. Delivered.

TTCN3 in Wireless Testing Eco Space

Agenda

- Challenges in Test environment development for Wireless Products
- Critical features of TTCN3 helping overcome the challenges
- Comparison of TTCN3 with other conventional languages
- Usage of TTCN3 across next generation wireless technologies

Challenges in Test Environment Development for Wireless Products

- ❑ To address conformance to defined standards and interoperability, we need a standardized architecture
- ❑ Environment which can be integrated with multiple hardware environments and multiple platforms
- ❑ Support for different encoding / decoding mechanisms
- ❑ Efficient display mechanisms
- ❑ Modular test architecture with flexibility to be adapted across various complex multi layered systems
- ❑ Run time environment should support timing constrained performance requirement
- ❑ Readable as well as machine byte level code
- ❑ Support of various debugging mechanisms
- ❑ Support for automation

Agenda

- Challenges in Test environment development for Wireless Products
- Critical features of TTCN3 helping overcome the challenges
- Comparison of TTCN3 with other conventional languages
- Usage of TTCN3 across next generation wireless technologies

Standardized Test Architecture

TTCN3 Test Architecture

- Standardized by ETSI
- Specifically designed for testing and certification
- Modular test architecture with flexibility to be adapted across different SUTs

Test Architecture Components

TM: Test Management, responsible for the overall test management

TL: Test Logging, responsible for maintaining the test logs

CH: Component Handling, responsible for distributing Parallel Test Components

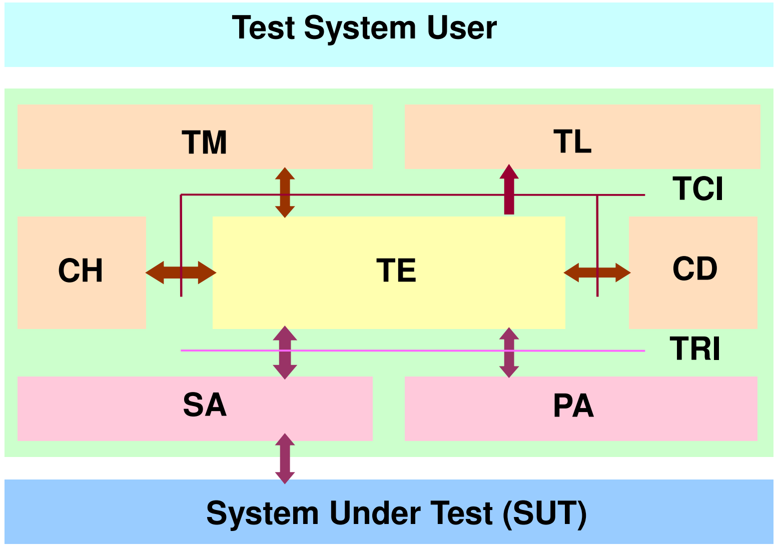
CD: Codec, responsible for encoding/decoding of the data associated with communication within TE

TE: TTCN3 Executable, responsible for interpretation and execution of TTCN3 ATS

SA: System Adaptor, responsible for communication between the Test System and System Under Test

PA: Platform Adaptor, responsible for implementation of external functions and timers

TTCN3 Test System



Test System User

TM

TL

TCI

CH

TE

CD

TRI

SA

PA

System Under Test (SUT)

Test Architecture Interfaces

TCI: TTCN3 Control Interface, specifies the interface between TM, TL, CH, CD and TE entities.

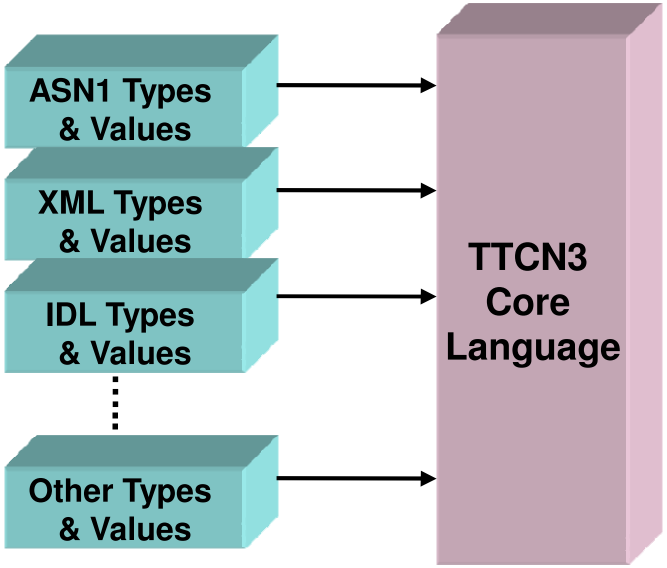
TRI: TTCN3 Runtime Interface, specifies the interface between TE, SA, PA entities.

Harmonization with other Languages, Hardware Environments

Harmonization with other Languages

TTCN3 can be integrated with test systems of other languages

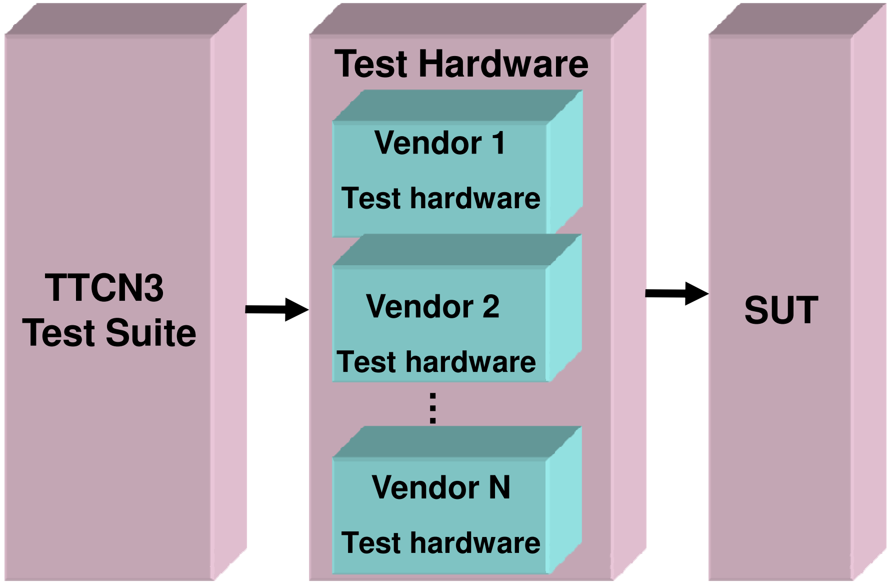
- Well harmonized with other languages like ASN.1, XML, IDL e.t.c.



Harmonization with other Hardware Environments

TTCN3 test systems are platform independent

- Test suites built on TTCN3 can be executed across various test hardware environments from multiple vendors

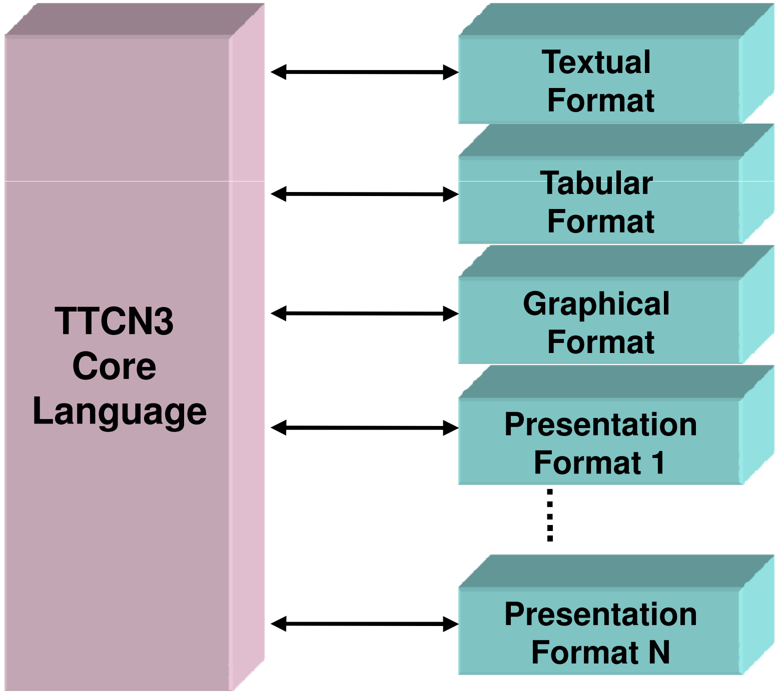


Efficient Display Mechanism

TTCN3 supports multiple presentation formats for e.g. Tabular and graphical. Depending on the application, specific presentation formats can be used to specify and visualize test cases

TTCN3 standard presentation formats

- Textual Format: Core language format
- Tabular Format (TFT): Similar to appearance and functionality to earlier versions of TTCN (Conformance testing oriented).
- Graphical Format (GFT): Provides an MSC based presentation format for UML or SDL oriented testing view. GFT can be used to graphically depict the functionality of
  - Control part of TTCN3 module
  - TTCN3 Test Case
  - Test Function
  - Alt step
- **Proprietary Formats:** TTCN3 can support other proprietary presentation formats as well



Dynamic Test Configuration and Efficient Communication Mechanism

TTCN3 test configuration consists of set of interconnected test components with well defined communication ports. Each configuration shall have only one MTC

Critical Features of TTCN3 Test Configuration

Dynamic Creation of Components

- Supports for dynamic instantiation of components and communication links

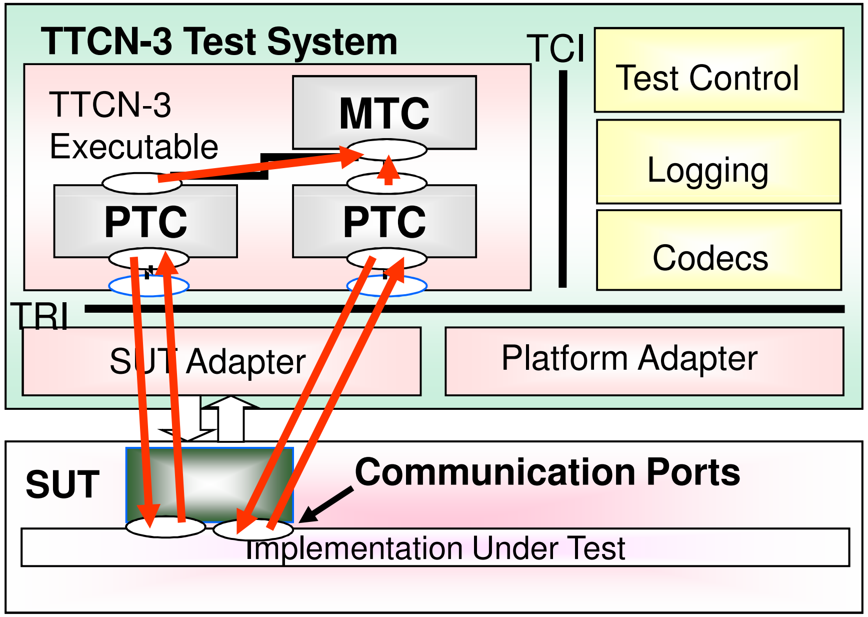
Communication Mechanism

- Test Components are connected to each other via communication ports
- Communication Ports are based on FIFO queue model
- TTCN3 provides support for synchronous as well as asynchronous communication mechanism

Timers

- Support for various timer operations like startTimer, stopTimer, timeout and check if timer is running e.t.c. Makes TTCN3 the best choice for testing language

TTCN3 Test Configuration



MTC: Process that runs the test case. Termination of MTC ends the test case execution

PTC: Components other then MTC are PTC. Termination of PTC does not terminate any other component

TTCN3 – The language

Critical Features

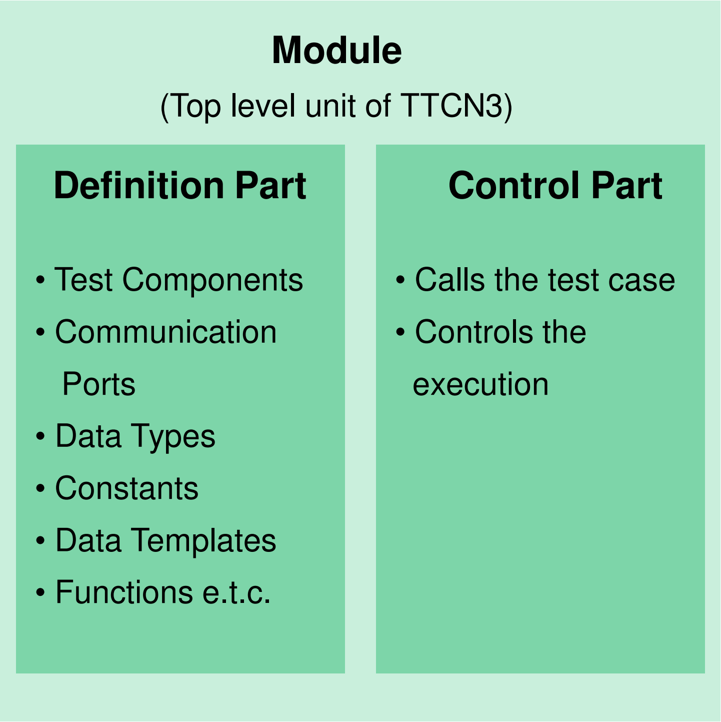
- ❑ **Syntax**
  - Well defined syntax
  - Support for basic data types, operators, templates e.t.c.
  - Concept of Test Verdicts
  - Built in matching mechanism
  - Support for external actions – Actions to be triggered from SUT
  - Support for message as well as procedure based communication
  - Look and feel of a programming language

All the above features provides an efficient and abstract way to specify the behavior of the test systems.

- ❑ **Compilation and Debugging options**
  - Not tied to any specific test execution environment, compiler or operation system
  - Provides debugging options as in any programming language

- ❑ **Automation**
  - Provides options to group test cases for automated execution
  - Support for PICS and PIXITS
  - Preamble, Postamble in test case body

Language Elements



Module

(Top level unit of TTCN3)

Definition Part

- Test Components
- Communication Ports
- Data Types
- Constants
- Data Templates
- Functions e.t.c.

Control Part

- Calls the test case
- Controls the execution

Agenda

- Challenges in Test environment development for Wireless Products
- Critical features of TTCN3 helping overcome the challenges
- Comparison of TTCN3 with other conventional languages
- Usage of TTCN3 across next generation wireless technologies

Comparison of TTCN3 with other Conventional Languages

	Flexible Test Verdicts	*Dynamic Configuration*	*Built in Matching Operations*	*Compile Time Error Detection*	*Defined Communication Mechanisms*	*Tracing Options*
TTCN3	🟢	🟢	🟢	🟢	🟢	🟢
Conventional Languages	🟠	🟠	🟠	🟠	🟠	🟠

🟢 **Built in Support**

🟠 **Support to be built by the user**

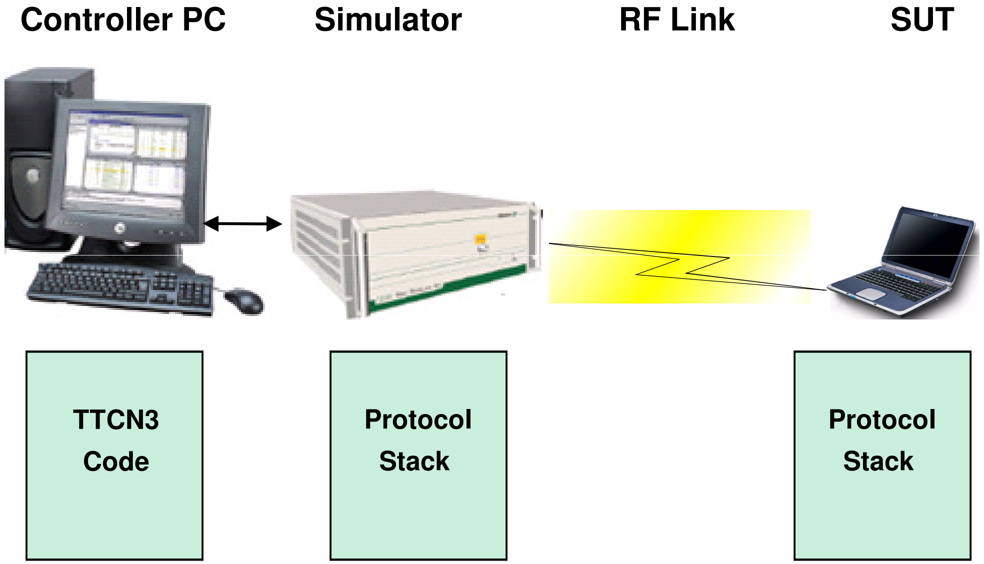
Agenda

- Challenges in Test environment development for Wireless Products
- Critical features of TTCN3 helping overcome the challenges
- Comparison of TTCN3 with other conventional languages
- Usage of TTCN3 across next generation wireless technologies

TTCN3 Test Set Up for Wireless Testing

Components of TTCN3 Test Setup

- **Controller PC**
  - TTCN3 code is used to simulate the test scenario
  - Connected to the simulator on a ethernet link
- **Simulator**
  - Simulates the Network entities for Handset testing
  - Simulated the handset for testing network entities
- **RF Link**
  - Wired RF used to connect the simulator to the handset
- **SUT**
  - May be either handset or network entity like BS



Controller PC **Simulator** **RF Link** **SUT**

TTCN3 Code

Protocol Stack

Protocol Stack

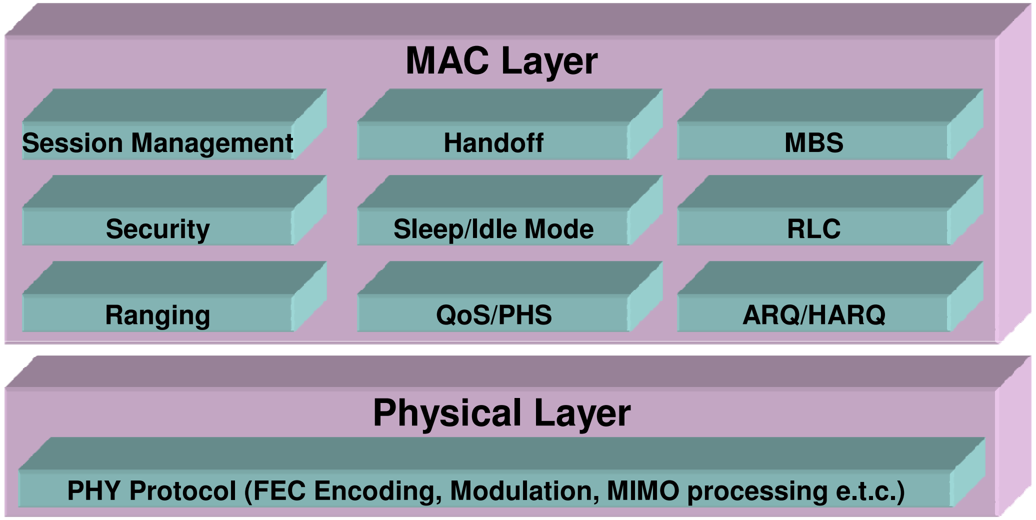
TTCN3 can be used to test various solution components across spectrum of technologies

- 2G/2.5G (GSM, GPRS)
- 3G (UMTS)
- 4G (WiMAX, LTE)

TTCN3 in WiMAX Test Architecture

WiMAX Air Interface Protocol Stack

- **MAC Layer**
  - Provides an interface between higher transport layers and physical layer
  - Responsible for controlling and multiplexing of data over PHY layer
- **Physical Layer**
  - Based on OFDM principle, designed for NLOS operations
  - Supports both TDD and FDD



MAC Layer

Session Management

Handoff

MBS

Security

Sleep/Idle Mode

RLC

Ranging

QoS/PHS

ARQ/HARQ

Physical Layer

PHY Protocol (FEC Encoding, Modulation, MIMO processing e.t.c.)

WiMAX TTCN3 Test Architecture

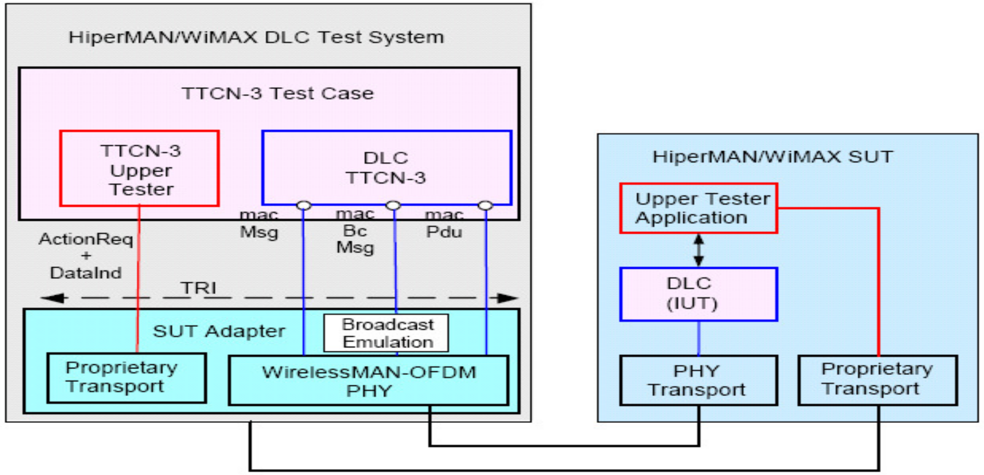
macMsg Port: Used to send and receive MAC management messages for Ranging, Connection set up.

macBcMsg Port: Used for MAC broadcast messages like DL/UL MAP, UCD, DCD.

macPdu Port: Used to send and receive for MAC PDU

Broadcast Emulation: Handles the sending and reception of broadcast messages

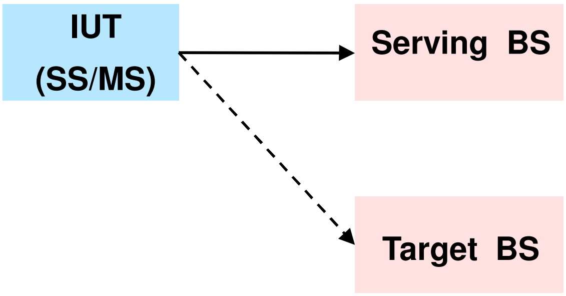
Upper Tester: Provides the option to trigger events from IUT



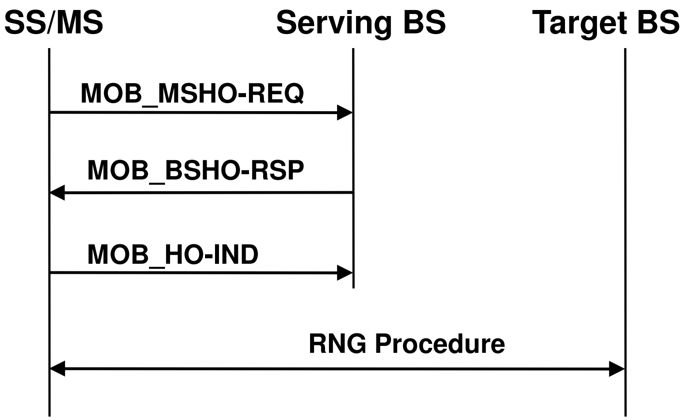
Source: ETSI TS 102 385-3

WiMAX Testing with TTCN3 – Hard Handover Scenario

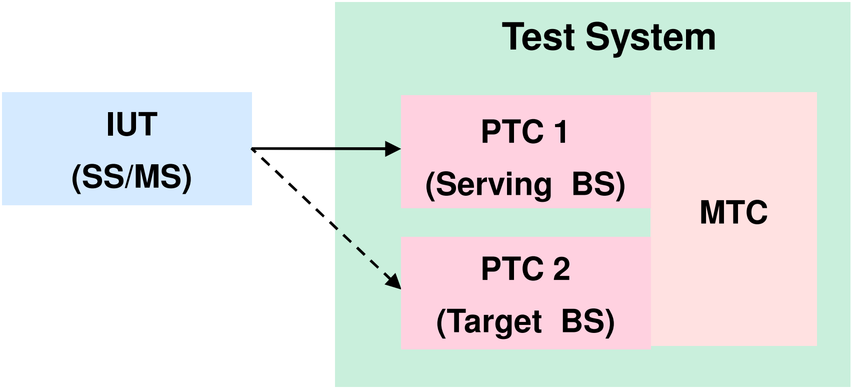
Configuration for Hard Handover



Call Flow for Hard Handover



TTCN3 Architecture for Hard Handover Testing



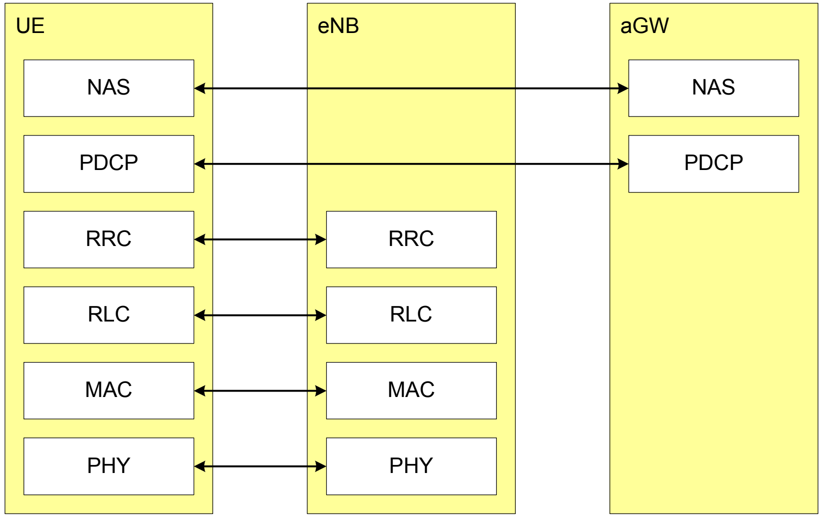
TTCN3 Advantages for Handover Testing

- Option to create multiple PTCs within the test case at run time to simulate the base stations required for Hard Handover
- Each of the simulated BS can be configured as Serving and Target BS for handling the Handover messages to and from the MS.

TTCN3 in LTE Test Architecture

LTE Protocol Stack

- RLC and MAC layers functions like scheduling, ARQ, HARQ.
- RRC performs the functions like Broadcast, Paging, Connection management, Radio Bearer control, Mobility functions, measurement reporting and control.
- PDCP layer performs the control plane functions like Header Compression, Integrity Protection and Ciphering.
- NAS performs functions like bearer management, Authentication, Paging, Security control.



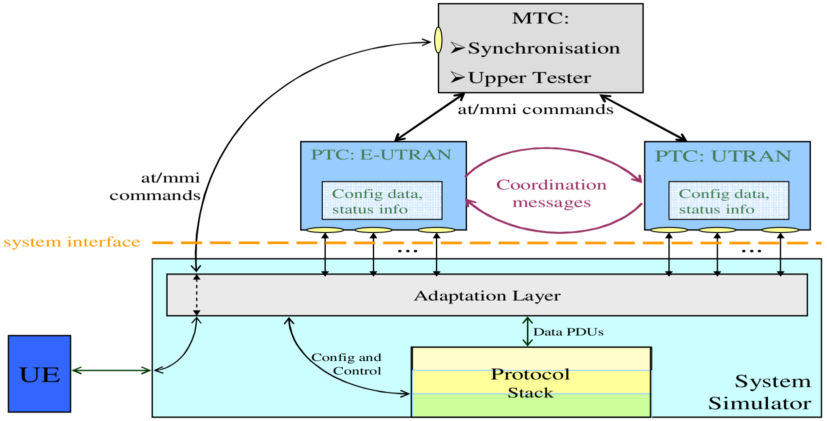
LTE TTCN3 Test Architecture

MTC - PTC: Common synchronization of PTCs and upper tester primitives.

MTC - System Interface: Upper tester primitives.

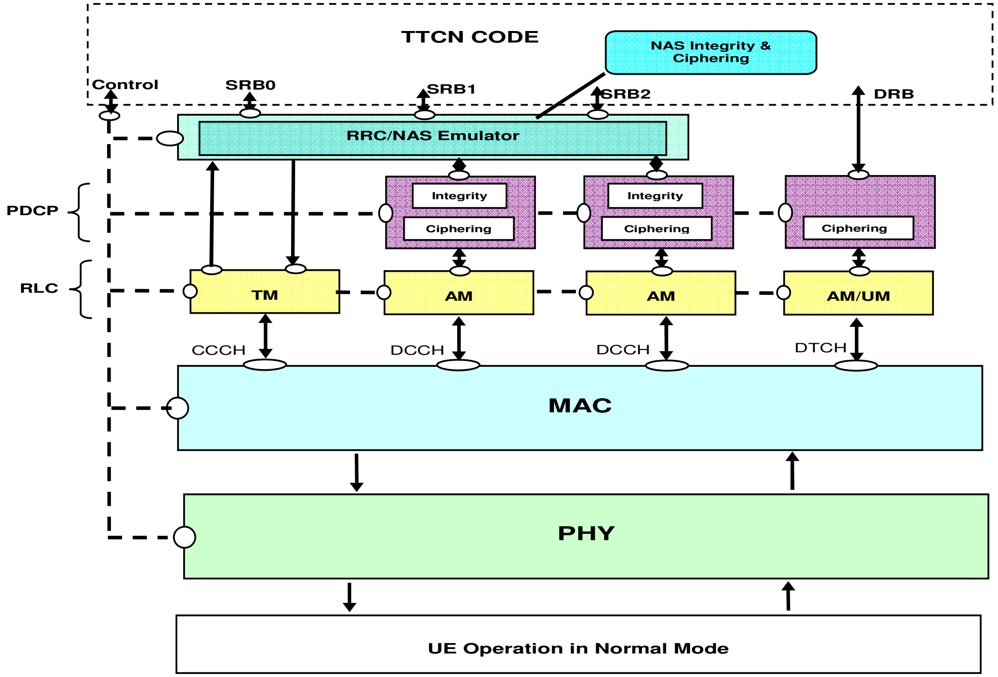
PTC - PTC: Primitives containing information for IRAT handover.

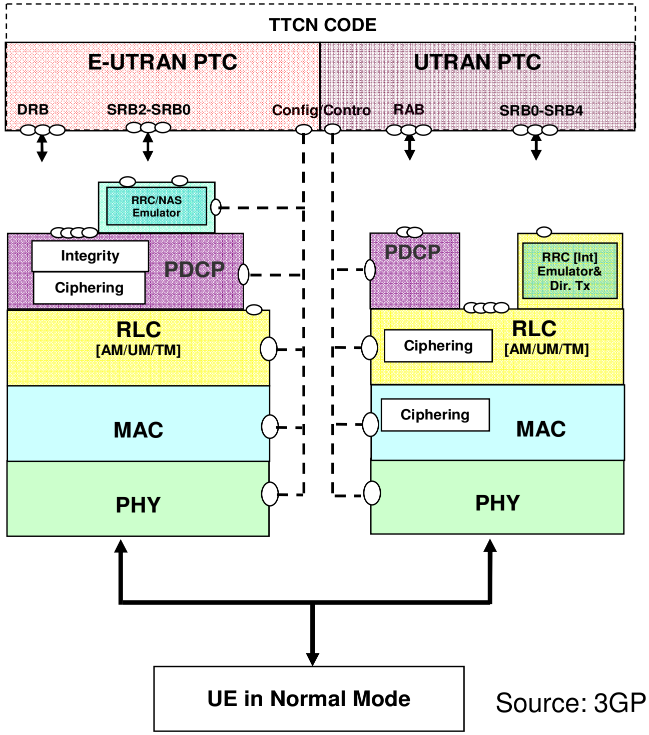
PTC - System Interface: Primitives containing peer-to-peer message and configuration primitives.



Source: 3GPP 36523-3

TTCN3 in LTE Test Architecture





Source: 3GPP 36523-3

LTE RRC testing model

- RRC/NAS is implemented as a parallel test component. Performs functions like broadcast, connection establishment, etc.
- The RRC/NAS emulator shall provide the Ciphering and integrity functionality for the NAS messages.

LTE Inter RAT E-UTRAN-UTRAN testing

- E-UTRAN and UTRAN functionalities will be in separate Parallel Test Components.
- The SS E-UTRAN part is as RRC testing.
- The SS UTRAN part consist of L1, MAC, RLC and PDCP

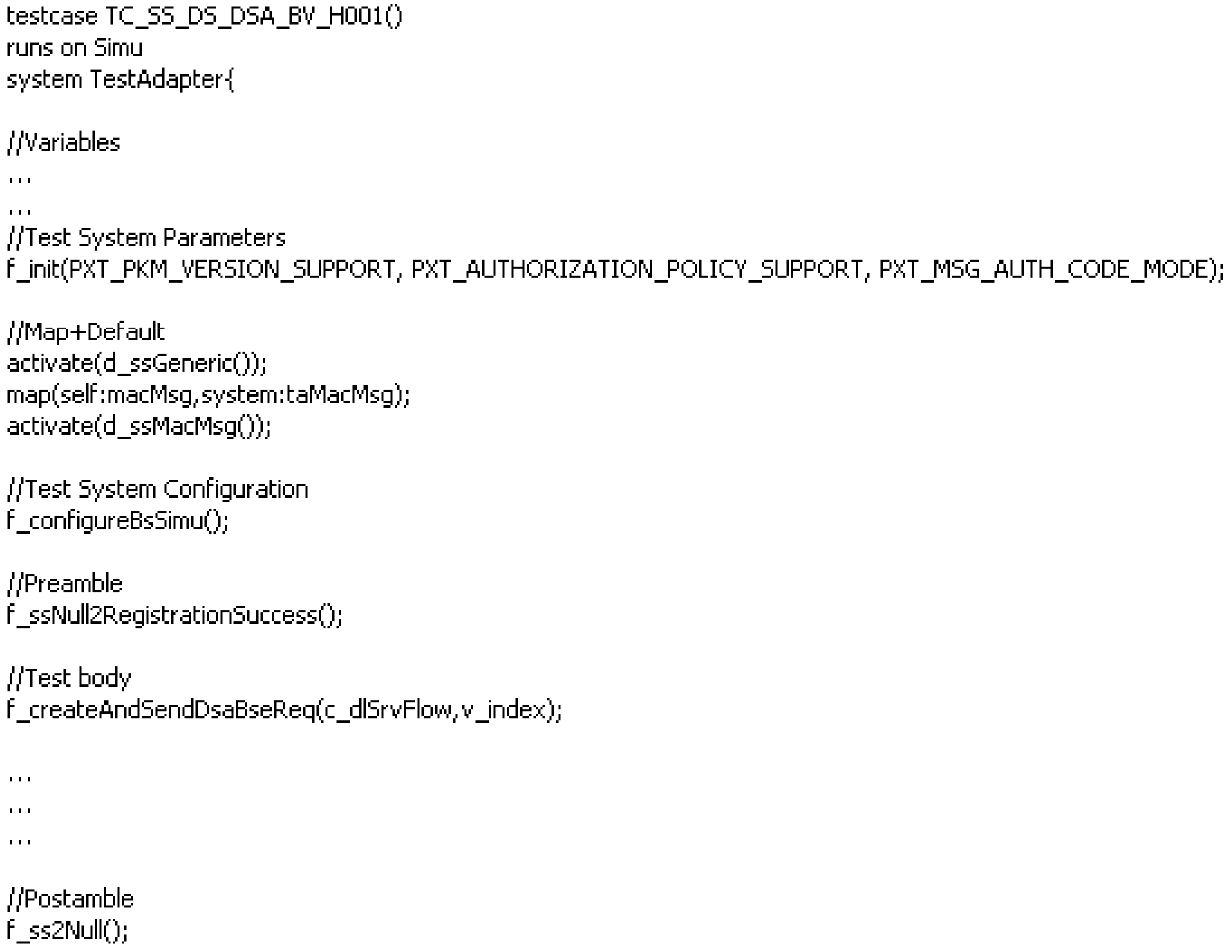
Thanks

APPENDIX

References

- TTCN3 Core Language: ES 201 873 – 1
- TTCN3 Tabular Presentation Format: ES 201 873 – 2
- TTCN3 Graphical Presentation Format: ES 201 873 – 3
- TTCN3 Operational Semantics: ES 201 873 – 4
- TTCN3 Run Time Interface: ES 201 873 – 5
- TTCN3 Control Interface: ES 201 873 – 6
- TTCN3 Integration with ASN.1: ES 201 873 – 7
- TTCN3 Integration with IDL: ES 201 873 – 8
- Conformance Testing for WiMAX – ATS: ETSI TS 102 545-3
- Air Interface for Broadband Wireless Systems: IEEE Std 802.16e™-2005
- Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Packet Core (EPC); User Equipment (UE) conformance specification: 3GPP TS 36.523-3

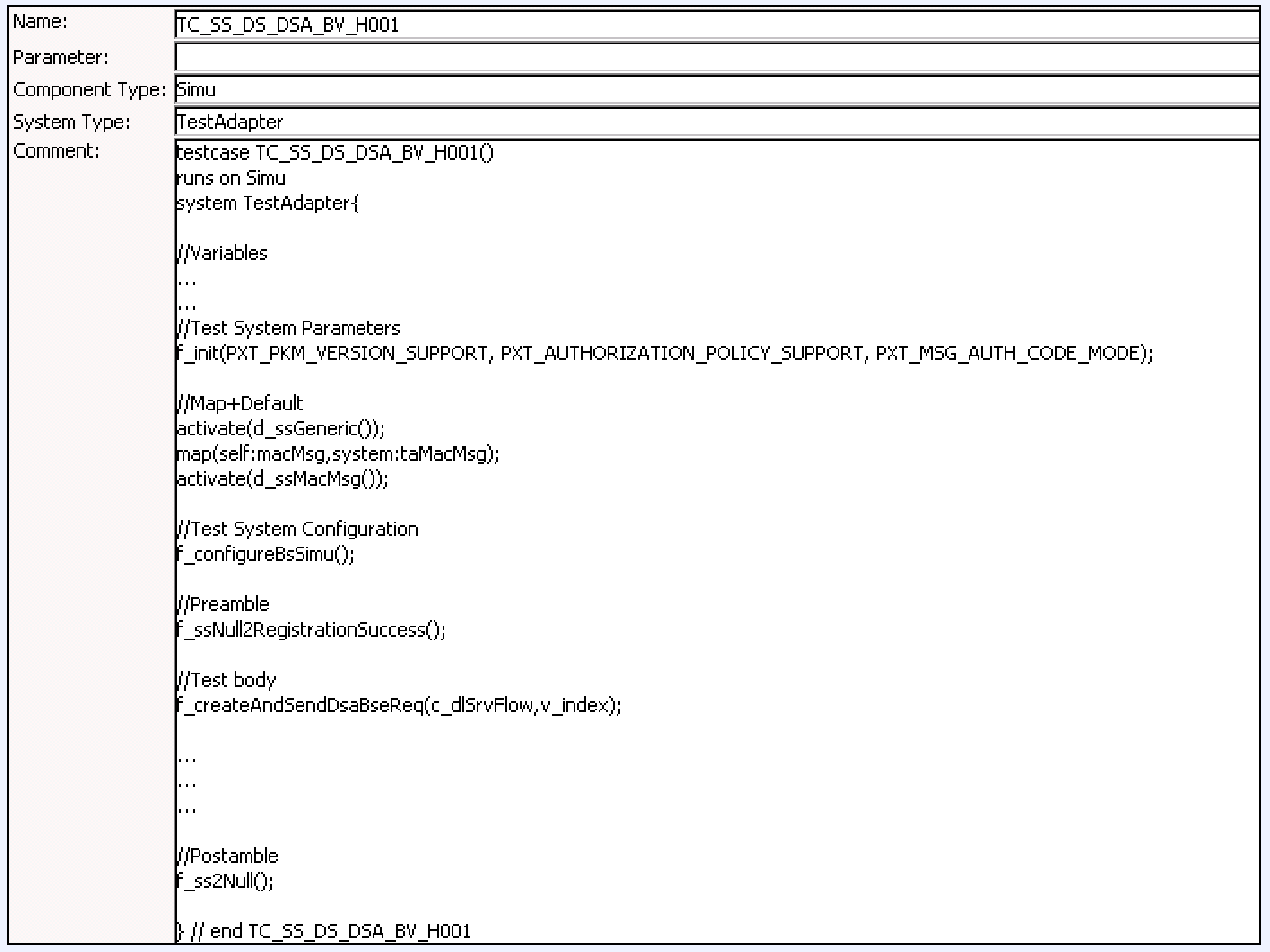
TTCN3 – Textual Format Presentation



```
testcase TC_SS_DS_DSA_BV_H001()
runs on Simu
system TestAdapter{

//Variables
...
...
//Test System Parameters
f_init(PXT_PKM_VERSION_SUPPORT, PXT_AUTHORIZATION_POLICY_SUPPORT, PXT_MSG_AUTH_CODE_MODE);

//Map+Default
activate(d_ssGeneric());
map(self:macMsg,system:taMacMsg);
activate(d_ssMacMsg());

//Test System Configuration
f_configureBsSimu();

//Preamble
f_ssNull2RegistrationSuccess();

//Test body
f_createAndSendDsaBseReq(c_dlSrvFlow,v_index);

...
...
...

//Postamble
f_ss2Null();
```

Back

TTCN3 – Tabular Format Presentation



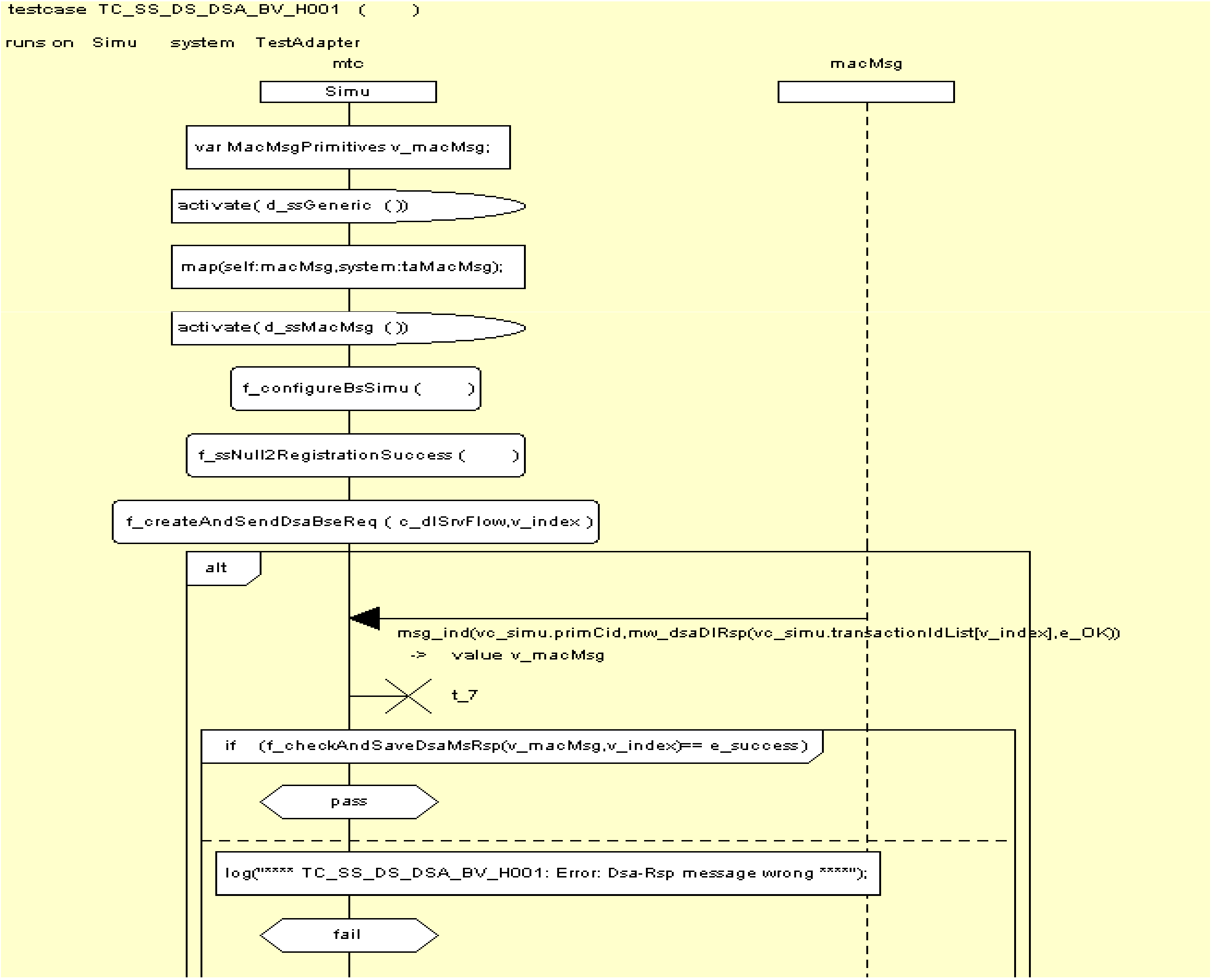
Name:	TC_SS_DS_DSA_BV_H001
Parameter:	
Component Type:	Simu
System Type:	TestAdapter

Comment:

```
testcase TC_SS_DS_DSA_BV_H001()
runs on Simu
system TestAdapter{

//Variables
...
...
//Test System Parameters
f_init(PXT_PKM_VERSION_SUPPORT, PXT_AUTHORIZATION_POLICY_SUPPORT, PXT_MSG_AUTH_CODE_MODE);

//Map+Default
activate(d_ssGeneric());
map(self:macMsg,system:taMacMsg);
activate(d_ssMacMsg());

//Test System Configuration
f_configureBsSimu();

//Preamble
f_ssNull2RegistrationSuccess();

//Test body
f_createAndSendDsaBseReq(c_dlSrvFlow,v_index);

...
...
...

//Postamble
f_ss2Null();

} // end TC_SS_DS_DSA_BV_H001
```

Back

TTCN3 – Graphical Format Presentation



```
testcase  TC_SS_DS_DSA_BV_H001  (       )
runs on   Simu     system   TestAdapter
mtc                                         macMsg
Simu
var MacMsgPrimitives v_macMsg;
activate( d_ssGeneric  ())
map(self:macMsg,system:taMacMsg);
activate( d_ssMacMsg  ())
f_configureBsSimu (        )
f_ssNull2RegistrationSuccess (        )
f_createAndSendDsaBseReq ( c_dlSrvFlow,v_index )
alt
    msg_ind(vc_simu.primCid,mw_dsaDlRsp(vc_simu.transactionIdList[v_index],e_OK))
     ->   value  v_macMsg
    t_7
    if   (f_checkAndSaveDsaMsRsp(v_macMsg,v_index)== e_success)
        pass
    - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
        log("**** TC_SS_DS_DSA_BV_H001: Error: Dsa-Rsp message wrong ****");
        fail
```

Back