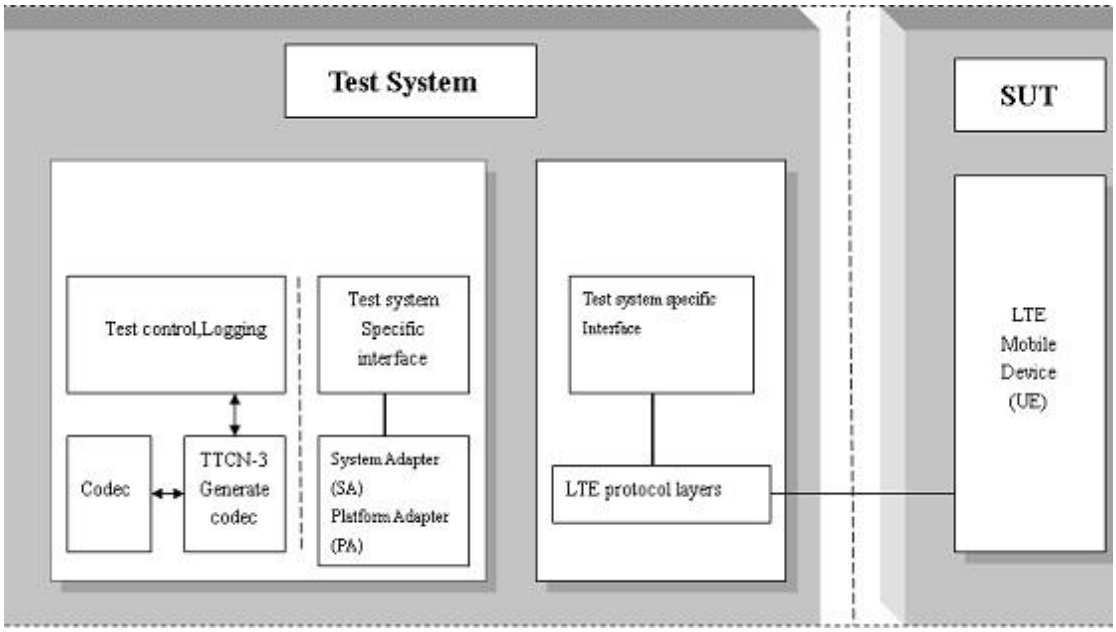
3GPP LTE 测试集结构介绍

LTE 测试集---测试系统概述

LTE 终端协议符合性是关乎互操作的大事情，3GPP 在制定 LTE 协议的同时也采用 TTCN-3 定义开发了 LTE 终端协议符合性测试集。该测试集是唯一被认可的 LTE 终端符合性测试集，也是被 GCF 等认证机构采用的认证检测测试集。本文可结合 TTCN-3/TTworkbench 培训培训内容。如下图所示是 LTE 测试系统概念性的测试架构。待测（SUT）是 LTE 移动设备（UE），测试系统有两个重要组件，主机和硬件适配。主机元素包括测试控制部分，测试执行引擎和相应的编解码。硬件适配器提供了到 SUT 的适配。在这个例子中意味着补齐 UE 接口的所有测试协议层以下的协议层。



图一、LTE test system architecture

LTE 测试集概览

一旦你在 TTworkbench 中打开 LTE 测试集开始浏览代码，你会发现 LTE 测试集不是一个简单的测试集。确实这个测试集非常复杂，不容易解释清楚所有细节。我们先提供一个概述帮助你理解总体的结构和测试概念。

在考量整个测试集的时候，你可能会发现这个测试集采用了我们在第 14 章和第 15 章中学习到的概念和建议。这个测试集采用了一个设计得非常好的

Framework.它也采用了命名惯例。例如，运行在一个组件上的 function 以 f 开头，altsteps 以 a 开头，发送模板以 CS 开头，接收模板以 CR 开头。除了清晰的命名，也定义了清晰的 declarations 布局，例如所有的变量都在功能前边定义，declarations 的顺序先是本地常量，接着是本地变量，然后是本地计时器。

LTE 测试集在最上层被分成了一系列的以 TTCN-3 为前缀名的文件夹。在每个文件夹里包含一个或多个 TTCN-3 模块，让我们先从第一个文件夹开始查看，这个文件夹以 TTCN-3 命名。当双击这个文件夹查看时，可以看到它包含两个 TTCN-3 模块：LTE_EPS_TS_SelectionExpressions 和 LTE_EPS_TS_Testcases.如果我们现在双击 LTE_EPS_TS_Testcases，源代码在 TTworkbench 中间窗口打开，这个模块可以看成是这个测试集的中心模块，它定义了所有的测试例。如果我们仔细查看代码内容，我们可以看到这个模块以一系列的导入 statements 开始，从其它模块导入了定义，然后它定义了所有的测试例，最后是控制部分，测试例是在控制部分被调用的。一个测试例是否被测试控制部分调用取决于 function f_ExecutionGuideline,在一些例子中结合了测试例子选择表达（Whether a specific test case is actually called when the control part is executed on the result from the function f_ExecutionGuideline in some cases combined with test selection expression)。

LTE 测试集---测试例定义(Test Case Definitions)

现在让我们通过一个测试例看看 LTE 测试例的构建以及组件的架构是如何组织的，如果我们看测试例 TC_6_1_1_1 （如下）

```
Testcase TC_6_1_1_1() runs on MTC system SYSTEM {
// @purpose
//     PLMN selection of RPLMN, HPLMN/EHPLMN, UPLMN and OPLMN
var GERAN_PTC    v_GERAN    := null ;
var EUTRA_PTC    v_EUTRA     :=EUTRA_PTC.creat alive ;
var UTRAN_PTC    v_UTRAN     :=null ;
timer t_GuardTimer := int2float(2400) ;
f_MTC_ConnectPTCs(v_EUTRA, v_UTRAN, v_GERAN ) ;
v-EUTRA.start   (f_TC_6_1_1_1_EUTRA()) ;
t_GuardTimer.start;
f_MTC_MainLoop(t_GuardTimer);
}
```

我们可以看到测试例运行在 MTC（master test component)上。注释定义的目的之后，我们可以看到三个变量的声明，提供可能的平行组件的参考。通过名字我们可以很容易的看到我们为测试中可能涉及用到的每个Radio access 技术定义了平行组件。GERAN_PTC 是 GSM 的，UTRAN_PTC 是 UMTS 的，EUTRA_PTC 是 LTE 的（虽然这套测试集是测试 LTE,但在有些测试中还是需要在测试系统和待测系统间创建通过其它无线技术的连接，如 Handover 测试）。在我们选择的这个例子中，我们实际上只需 LTE 连接，因此只有 LTE PTC 我们使用了 create 创建命令语句。在创建完平行组件后，测试例定义了计时器，然后调用功能函数 f_MTC_ConnecPTs.这个功能函数创建了 MTC 和 PTCs 之间的所有必要的连接。在创建完组件并进行了必要的组件连接后，我们现在实际开始要求的测试行为，在传参函数 function f_TC_6_1_1_1_EUTRA 传递的 LTE PTC 上采用 Start 操作。采用 StarThis is done using the **start** operation on the LTE PTC passing in the function f_TC_6_1_1_1_EUTRA.就是在这个函数中，实际的测试行为，与 SUT 发送和接收消息被定义。我们稍后还要回到这个函数。接下来测试例启动 guard timer ,然后调用行数 function f_MTC_MainLoop ，f_MTC_MainLoop 函数的基本上就是只等待平行测试组件完成行为。

从这个例子中我们可以看到 LTE 测试例的一些特点，实际上如果你查看整个模块，你会发现这些特点在所有例子中是共通的。首先我们可以看到顶层测试例只在 MTC 执行，其次我们可以看到测试例的实际行为执行在平行组件上。所有测试例运行在 MTC 上是相同的任务：创建平行组件 PTCs，连接平行组件 PTCs,开始计时器（guard timer）,然后等待平行组件完成。

LTE 测试集---测试行为定义 Test Behaviour Definition

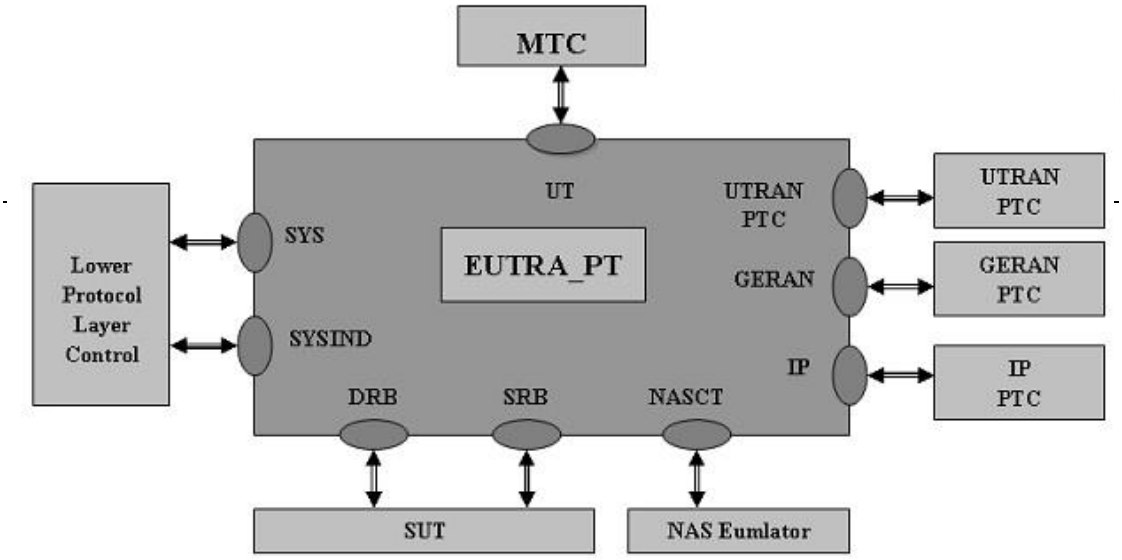
现在让我们来看这个测试例的实际的测试行为定义，测试行为定义在 function f_TC_6_1_1_1.这个 function 定义可以在 ttcn-3\6_1 文件夹的 module Idle_PLMNSelection 中找到。这个函数中的行为可以被分为三部分。第一部分是 preamble,使待测设置在执行测试例需要的状态。然后是测试主体也就是我们要测试的行为，最后是初始化待测到原来的状态，以便后边的测试例测试使用。找到这些变量哪里开始和哪里结束的最简单方法是调用 function f_EUTRA_TestBody_Set。这个函数在测试最开始和测试结束部分被调用，在开始部分参数是 true,在结束部分参数是 false。测试体本身被分成一系列的 steps，这些测试步骤对应与 3GPP 标准：终端协议符合测试说明【（3GPP TS 36.523-1 V8.3.0(2010-06) 3rd Generation Partnership Project:Technical Specification

Group Radio Access Network;Evolved Universal Terrestrial Radio Access(E-UTRA) and Evolved Packet Core (EPC);User Equipment(UE) Conformance Specification Part 1:Protocol Conformance Specification (Release 8)描述的测试。】

LTE 测试集---EUTRA 平行测试组件（EUTRA Parallel Test Component）

现在让我们来研究运行函数的平行测试组件（parallel test component）。如我们在 4.10 章节中学习的，平行组件类型要与 Runs on 放在一起用。从 f_TC_6_1_1_1(如下）我们可以看到组件类型是 EUTRA_PTC.这个类型在 ttcn3/commonEUTRA 文件夹的 EUTRA_Component 模块中定义。一般来讲端口可以分成四组，在图二的上边是对 MTC 的接口，这个 UT 接口是控制平行组件上行为的执行。右边的这组端口是与其它平行测试组件之间协调和交流的接口。左边端口是与 TTCN-3 执行引擎下协议栈和仿真器的接口，最后在图形下边这组端口负责与待测通信。SRB 控制平面信息端口，DBR 是用户平面数据端口。控制平面是 LTE 核心网络和移动设备信令流量的通道。用户平面是所有用户数据的信息通道，也就是移动设备用户实际想要发送和接收的信息。

```
Type component EUTRA_PTC{
Var     EUTRA_Global_Type vc_EUTRA_Global;
Port    EUTRA_SYSTEM_PORT       SYS;
Port    EUTRA_SYSIND_PORT       SYSIND ;
Port    EUTRA_SRB_PORT          SRB ;
Port    EUTRA_NASCTRL_PORT      NASCTRL ;
Port    EUTRA_DRB_PORT          DRB ;
Port    IRAT_CO_ORD_PORT        UTRAN ;
Port    IRAT_CO_ORD_PORT        GERAN ;
Port    IP_RAT_CTRL_PORT        IP ;
} ;.
```



图二、EUTRA Component Type

LTE 测试集---测试集模块结构

已经研究了测试例 TC_6_1_1_1 以及相关的函数 f_TC_6_1_1_1,现在是时候更好的理解测试集全部模块的结构了。我们可以看到包含平行测试组件测试行为的模块被分成了若干文件夹，体现了 3GPP TS 36.523-1 V8.3.0(2010-06)一致性测试描述部分的结构和编号,这些文件夹的名字形式都是 ttcn3\n 或者 ttcn3\n_n, 这里 n 是一个数字。这些组包含的测试领域如下：

Table 16.3 Test group areas for LTE suite

Group	Sub-group	Test area
6	-	Idle mode operations
	6.1	In a pure E-UTRAN environment
	6.2	Muti-mode environment (E-UTRAN,UTRAN,GERAN,CDMA2000)
	6.3	Closed subscriber group cells
7	-	Layer 2
	7.1	MAC
	7.2	RLC
	7.3	PDCP
8	-	RRC
	8.1	RRC connection management procedures
	8.2	RRC connection reconfiguration
	8.3	Measurement configuration control and reporting
	8.4	Inter-RAT handover

	8.5	RRC others
9	-	EPS mobility procedures
	9.1	EMM common procedures
	9.2	EMM specific procedures
	9.3	EMM connection management procedures(S1 mode only)
	9.4	NAS Security
10	-	EPS session management
11	-	General tests
12	-	E-UTRA radio beaer tests
13	-	Multi layer procedures

LTE 测试集-RRC 层协议消息定义

这套测试集测试的最主要协议是 RRC(Radio Resource Control)，RRC 是 LTE 的层 3 协议，RRC 被 LTE 测试集中大多数的 TTCN-3 消息采用与待测交换信令消息，RRC 协议是采用 ASN.1 定义的，如在第 9.5.1 章节中描述的，ASN.1 的类型可以导入并在 TTCN-3 定义中直接采用，回到 TTCN-3\6_1 文件夹的 Idle_PLMENSelection 模块，我们可以在导入的最开始部分看到 **Import from** EUTRA_RRC_Definitions **language** “ASN.1 :1997” **all** ; 这个语句导入了 RRC 消息类型定义。应用 TTCN-3 工具，我们可以直接看这些定义。ASN.1 模块可以在 ttcn3\comonEUTRA_Defs 文件夹中找到，模块的文件名字是 EUTRA_RRC_ASN1_Definitions.asn.在这个问价夹中还有另外一个名字相同的模块，但是文件扩展名为 ttcn3view。这个模块使我们可以看到翻译为 TTCN-3 语句后的 ASN.1 定义。

在文章最后的部分，让我们来看看 ASN.1 数据类型定义是如何在测试例中被应用的。让我们在回到文件夹 ttcn3\6_1 模块 Idle_PLMNSelection 的函数 f_TC_6_1_1_1。看一下测试主体部分的 step 3.

```
//   Step 3: Receive RRCConnectionRequest on Cell 12
SRB.receive(car_SRB0_RrcPdu_IND(eutra_Cell12,
                                Cr_508_RRCConnectionRequest));
T_IdleMode_GenericTimer.stop;
```

//* @verdict pass RRCConnectionRequest message receive on Cell 1
F_EUTRA_PreliminaryPass(_FILE_, _LINE_,"Test case 6.1.1.1 step 3");

我们可以看到这步是以 receive statement 开始的.从我们之前对 EUTRA_PTU 组件类型的了解，我们知道因为 receive statement 是在 SRB 端口执行，它正在期待从待测 SUT（The LTE mobile device）收到信令消息。期待接收到的消息是用 cas_SRB0_Rrcdu_REQ 模板定义的，如果我们在 ttcn3\commonEUTRA_Template 文件夹中的 EUTRA_SRB_Templates 模块看一下这个模板的定义，我们可以看到我们实际期待接收到的消息类型是 SRB_COMMON_REQ。我们也注意到模板和模板使用的参数应用到我们在 11.6 章节中学习到的 Template restrictions 原则。The(Value) notation 指出模板必须解析到一个特定的值。

Example SRB template

Template (value) SRB_COMMON_REQ cas_SRB0_RrcPdu_REQ(
CellId_Type p_CellID
Template(value) TimingInfo_Type p_TimingInfo,
Template (value) DL_CCCH_Message p_RrcPdu) :=
Common := cs_ReqAspCommonPart_SRB(p_CellID,tsc_SRB0,p_TimingInfo),
Signalling := {
Rrc :={
Ccch :=p_RrcPdu
},
Nas := **omit**
}
};

SRB_COMMON_REQ 类型的定义可以在文件夹 ttcn3\CommonEUTRA_Defs 中的 EUTRA_ASP_Srbdefs 模块中找到。如下 SRB_COMMON_REQ type definition

Type record SRB_COMMON_REQ{
/* common ASP to send PDUs to SRB0, SRB1 or SRB2 * /
ReqAspCommonPart_Type Common,
C_Plane_Request_Type Signalling
};

从这个类型定义，我们可以看到消息被分成了两部分：Abstract Service Primitive（ASP）信息和信令。在 LTE 测试集中，所有的消息都被分成了 ASP 和 Protocol Data Unit(PDU)。待测实际要收到的消息是在 PDU 中被定义的，在这个例子中包含在信令领域里。ASP 和与之相关的的任何相关的信息字段是纯粹的为 TTCN-3 测试执行引擎下的协议栈和适配层。

如果我们想找到 ASN.1 定义是在哪里被应用的，我们需要仔细看这个类型的 PDU 部分，在我们的这个例子中应该解析到 RRC 消息。其中包含的 PDU 的信令字段的类型是 C_Plane_Request_ Type。它的类型定义可以在同一模块 SRB_COMMON_REQ.In 中找到。如下 C_Plane_Request_Type definition

```
Type record C_Plane_Request_Type {
/* RRC and/or NAS PDU to be send to the UE */
RRC_MSG_Request_Type      Rrc             optional,
NAS_MSG_Requestlist_Type  NAS             optional
} ;
```

这个测试案例，其中涉及的 RRC prtotocol 这只是个第一个域 Rrc，它是相关的。这个域有 RRC_MSG_Request_Type 类型，可以在 ttcn3\CommonEUTRA_Defs 文件夹的 EUTRA_CommonDefs 模块中找到这个类型的定义。如下 RRC_MSG_Request_Type definition

```
Type Union RRC_MSG_Request_Type{
/* DL RRC PDU on CCCH or DCCH * /
    DL_CCCH_Message          Ccch,
    DL_DCCH_Message          Dcch
};
```

看着这个类型，我们终于有了到 ASN.1 定义的链接。如果我们看这个类型的第一个域，Ccch 是 DL_CCCH_Message 类型中的一个域，这个类型是在我们之前了解过的模块 EUTRA_RRC_ASN.1_Definitions 中定义的。