[1. 框架 2](#_Toc393894500)

[1.1 总体框架 2](#_Toc393894501)

[1.2 测试工具框架 2](#_Toc393894502)

[2. 测试工具与RF间接口 2](#_Toc393894503)

[2.1 承载 2](#_Toc393894504)

[2.2 接口消息 3](#_Toc393894505)

[2.2.1 接收测试脚本 3](#_Toc393894506)

[2.2.2 启动测试 3](#_Toc393894507)

[2.2.3 事件通知 4](#_Toc393894508)

[2.2.4 变量通知 4](#_Toc393894509)

[2.2.5 测试报告 4](#_Toc393894510)

[2.2.6 测试结束 5](#_Toc393894511)

[2.2.7 异常终止 5](#_Toc393894512)

[3. 测试脚本 5](#_Toc393894513)

[3.1 发送消息 5](#_Toc393894514)

[3.2 接收消息时的响应 5](#_Toc393894515)

[3.3 事件发送和接收 6](#_Toc393894516)

[3.4 等待 6](#_Toc393894517)

[3.5 测试结束 6](#_Toc393894518)

[4. 知识库 6](#_Toc393894519)

[4.1 PSTT 6](#_Toc393894520)

[4.1.1 消息默认值 6](#_Toc393894521)

[4.1.2 默认流程处理 6](#_Toc393894522)

[4.1.3 存储格式 6](#_Toc393894523)

[4.1.4 管理界面 6](#_Toc393894524)

[4.2 其他工具 6](#_Toc393894525)

[5. 模拟网元配置 6](#_Toc393894526)

[6. 测试用例的执行 7](#_Toc393894527)

[6.1 开始执行 7](#_Toc393894528)

[6.2 默认行为 7](#_Toc393894529)

[6.3 测试用例指定的行为 7](#_Toc393894530)

[6.4 执行结束 7](#_Toc393894531)

[6.5 异常 7](#_Toc393894532)

# 框架

## 总体框架



1. 不同的测试工具与RF间采用统一接口；
2. 测试工具与RF间接口的消息与业务无关，业务的描述在另一个层面上；

## 测试工具框架



1. RF会根据模拟网元，将测试脚本分解开，分别发送给各个不同的模拟网元
2. 测试工具中，每个模拟网元的执行引擎使用独立的实例，互不干扰；；
3. 模拟网元间的协作通过事件通知实现，同一个网元内的事件通知，可以考虑使用共享内存等；
4. 模拟网元间需要共享的数据，通过变量通知实现，同一个网元内的变量通知，可以考虑使用共享内存等；

# 测试工具与RF间接口

## 承载

测试工具与RF间，采用TCP Socket方式连接。原因如下：

1. Socket是操作系统提供的编程接口，稳定可靠；
2. TCP采用可靠传输方式，不需要考虑数据的重传、速率适配等问题。

每个测试工具守护一个Socket，用于与RF间相互通信，执行脚本分发、事件通知和测试报告等，供该工具模拟的所有网元公用。

## 接口消息

### 接收测试脚本



RF通过Send Script将脚本发送给测试工具中的指定网元。需要携带模拟网元名称和模拟网元要执行的脚本。

网元名称和测试脚本均为文本信息。测试脚本中包含一个测试中所有需要执行的消息。消息的格式参见章节5。

Send Script消息需要测试工具的响应来证实脚本已经分发完成，只有得到确认后，才可以启动测试过程。响应消息中，需要携带模拟网元名称。

### 启动测试



RF通过Start消息来通知模拟网元启动测试，模拟网元将按照收到的测试脚本执行测试过程。收到Start后，模拟网元需要响应该消息，以便通知RF测试脚本已经启动执行，这是因为触发业务流程的脚本需要最后执行，并要能够保证其他模拟网元的脚本已经在运行状态。

启动测试消息和响应中，需要携带模拟网元名称。

### 事件通知



测试工具执行脚本过程中，依照脚本触发事件。RF收到事件后，将事件广播给所有的其他测试工具。测试工具发出的事件通知中，携带触发事件的模拟网元和事件内容。RF广播出去的事件中，携带事件源、模拟网元名称以及事件内容。

事件通知不需要响应。

### 变量通知



变量通知的目的是在测试工具间共享变量。测试工具执行脚本过程中，依照脚本对变量赋值时，触发该过程，想RF发送变量通知，RF将变量通知广播给所有其他的测试工具，各个测试工具收到该通知后，将变量的变更保存在各自的数据区中。

变量通知消息中，需要携带变量来源、变量名称和变量值。

变量通知消息不需要响应。

### 测试报告



当测试脚本执行结束后，各个模拟网元需要向RF上报测试报告，包括测试过程中的各种检查结果，由RF来判定测试的成功与失败。

测试报告采用文本形式，不需要响应。

### 测试结束



识别出测试结束的测试工具向RF发送End消息，RF收到后终止所有其他测试工具的脚本执行。

End、Stop、Stopped消息中需要携带目标模拟网元名称。

### 异常终止



当测试过程中发生异常，无法继续执行时，测试工具向RF发送异常消息，RF收到后会判定本次测试失败，并终止测试。

异常消息中需要携带异常发生的模拟网元和异常的原因。异常消息不需要响应。

# 测试脚本

## 发送消息

模拟工具执行这条脚本时，按照脚本描述发送指定消息到被测网元。脚本中包括消息的源、目的、名称和参数。

## 接收消息时的响应

接收消息的响应用于控制测试工具的定制化动作，当收到某消息时应该如何响应。

接收消息时的响应包括要接收的消息和响应内容。

其中要接收的消息包括消息的源、目的、名称和参数，响应内容可以是各种其他脚本元素，但不包括嵌套的接收响应和脚本结束条件。一个响应中可能包括一组动作，直到下一个接收响应或脚本结束前的内容，都属于这个接收的响应。

## 事件发送和接收

事件通知用于不同测试工具间的协作。只有无法通过DUT的消息流程来协调工具动作时，才需要使用事件通知。事件发送包括事件源和事件内容，事件接收包括事件源、目的和事件内容。

工具执行事件发送时，向RF发送事件后继续执行；工具执行事件接收时，必须阻塞等待事件发生后，才能够继续执行。

## 等待

这一脚本用于控制工具等待若干时间后继续执行下一步。

等待消息中包括要等待的模拟网元和要等待的时间。

## 测试结束

当测试工具执行到测试结束后，向RF上报测试报告，并发送测试结束消息，通知RF结束测试过程，RF向所有测试工具发送结束消息，终止测试。

# 知识库

## 概述

在测试脚本中，并需要指明测试中的所有消息序列，只需要指明测试关注的内容即可，其余的执行，需要测试工具自行完成。

在测试用例没有明确的情况下，测试工具需要完成两项工作：如何响应收到的消息，如何填写消息的内容。为了解决这两个问题，测试工具中需要设计知识库，用于在测试用例没有说明情况下，确定默认的行为模式。

对应上面的两个问题，知识库分为两个部分，一是如何响应收到的消息，另一个是各种消息的默认值。

## 消息默认响应

当测试工具收到DUT消息时，首先按照脚本中的描述来响应该消息，如果在脚本中没有描述，就需要按照知识库中该消息的默认方式来响应该消息。

知识库中应该保存所有需要响应的消息，并定义其默认响应方式。

## 消息默认值

当测试工具需要发送消息时，该消息的参数首先按照脚本的描述来填写，对于脚本中没有填写的参数，则需要按照知识库中的消息默认值来填写。

在使用默认响应时，也不需要描述所有参数，因此当默认响应中有消息参数没有描述时，也需要使用知识库中的消息默认值来填写。

## 管理界面

知识库需要根据待测试的内容来构建，因此需要提供人机界面对知识库中的内容进行管理，包括添加、修改、删除等。

# 模拟网元配置

模拟网元的配置沿用现有PSTT的配置方式。

# 测试用例的执行

## 启动

测试工具启动后，就为各个模拟网元启动执行引擎，联通与DUT的通信通路，并守护与RF的接口，等待执行测试脚本。

## 接收测试脚本

RF将测试脚本通过Socket传送给测试工具，测试工具根据脚本归属的网元，将脚本传递给各个执行引擎，准备开始执行测试。每个脚本收到后，都需要向RF确认脚本已经收到，以便RF能够知晓启动测试的时机。



## 开始执行

RF向测试工具发起开始执行消息后，测试工具中的模拟网元就根据已经收到的脚本开始执行。每个模拟网元都会收到RF发送的开始执行消息，因此不需要工具关注模拟网元间的启动顺序。每个模拟网元的执行相互独立，互不相干，执行开始的顺序由RF来控制。



## 默认行为

测试脚本中仅描述用户关注的行为，测试过程中，会有大量的消息收发和参数在脚本中没有描述，这时，测试执行引擎需要查询知识库。在接收到消息时，查找默认的响应方法，在发送消息时，查找消息中各个参数的默认值。这样，当测试脚本没有指定如何操作时，工具也能够正确的执行下去。





## 测试用例指定的行为

与默认行为相对应的，测试过程中有些行为是用户关心的，希望能够控制。这时，就需要按照测试脚本中描述的行为来执行。测试用例中指定的行为也分为两种，一是接收消息时如何响应，另一个是发送消息时，参数如何填写。这两种情况都需要优先按照用户在测试脚本中描述的来填写。

此外，用户描述的消息参数只是所有参数的子集，其余参数的填写也需要参照知识库来执行。

测试用例指定的行为仅在本测试用例中有效，当测试终止后，所有测试用例指定的行为将不再生效。

流程参见上一节流程图。

## 事件

当测试期望的模拟网元间行为无法通过DUT的消息序列协调时，需要使用事件通知来控制模拟网元间的执行顺序。事件的等待者会阻塞测试用例的执行，直到事件发生或超时，事件的发起者会立即发起事件，继续执行。

测试脚本中会编写事件的发起方和等待方。发起方会触发事件，由RF广播给所有模拟网元。对于同一个工具内部的模拟网元，事件通过进程间通信分发。

等待事件超时将导致测试用例异常结束。

## 变量

变量用于在模拟网元间共享数据。在测试脚本中，可以把消息中的某些参数书写为变量的形式。接收消息中的变量将被保存，发送消息中的变量将引用该变量的值。变量是所有模拟网元共享的，因此当变量新建或改变时，需要通过RF将变量的值广播给所有的模拟工具。

变量的广播方式与事件类似，发生变量改变的模拟网元向RF发送变量通知，RF向其他模拟工具广播变量通知。同一个工具内的模拟网元间，不需要通过RF广播，采用内部通信即可。

变量仅在一个测试用例内有效，测试用例结束后，所有变量全部清除。

## 执行结束

测试脚本中，必定会在一个模拟网元上编写测试结束。当模拟网元执行到测试结束时，向RF上报测试报告，然后结束本模拟工具中所有模拟网元的测试执行，各个模拟网元上报测试报告后终止。RF通过广播的方式，通知其他模拟工具结束测试执行，各个模拟网元上报测试报告后终止。

执行结束后，测试工具恢复初始状态，已经执行的测试脚本、变量、事件等全部初始化。

## 异常

测试过程中发生各种异常，如脚本错误、知识库中没有数据、等待超时等，全部通过异常上报给RF，其中需要携带异常发生的原因。上报异常后，测试脚本终止本次执行，恢复初始状态。

RF收到异常后，结束所有测试工具的执行，并判定本测试用例执行失败。