使用XML开发脚本

作者：刘雪松

联系方式：18115503264

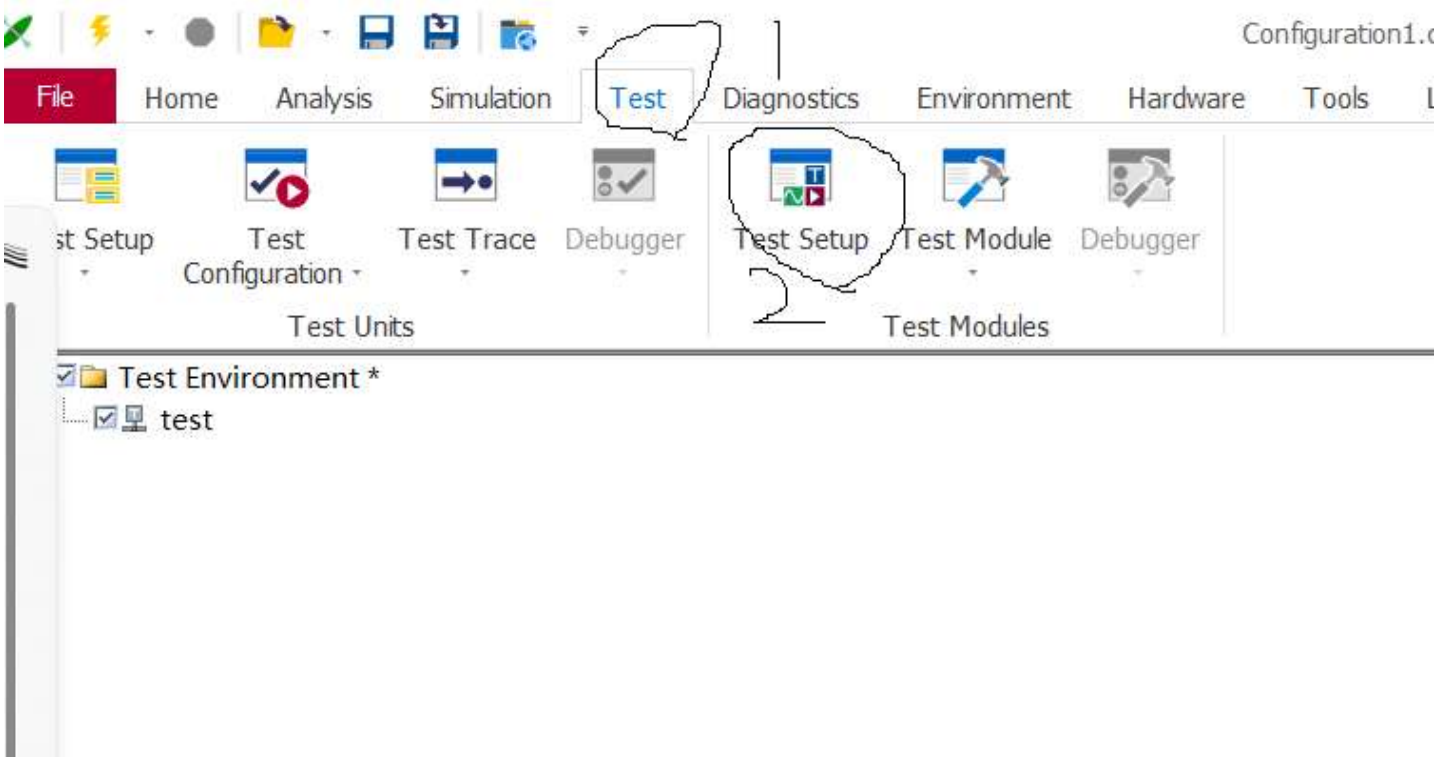
Revision history

序号	版本	变更内容	变更人	日期
1	1.0	First release	Liu xuesong	2024/4/21

目录

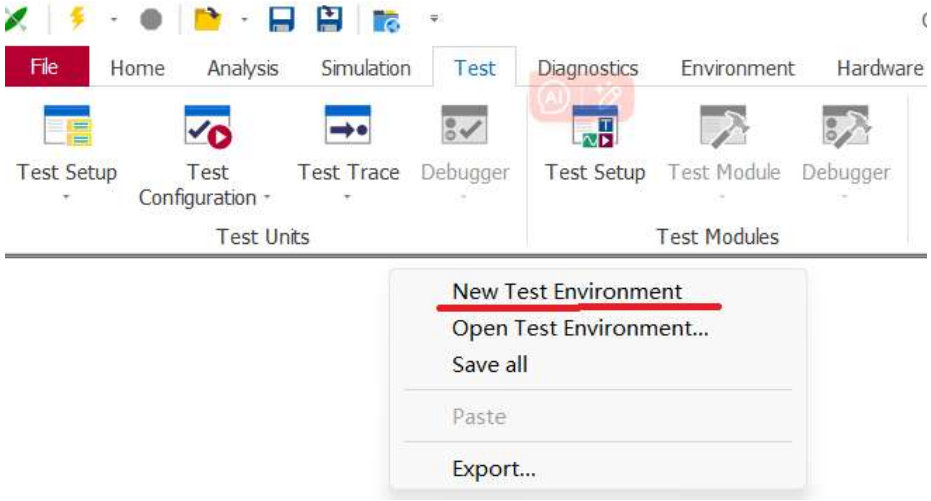
常用的canoe自动化开发脚本

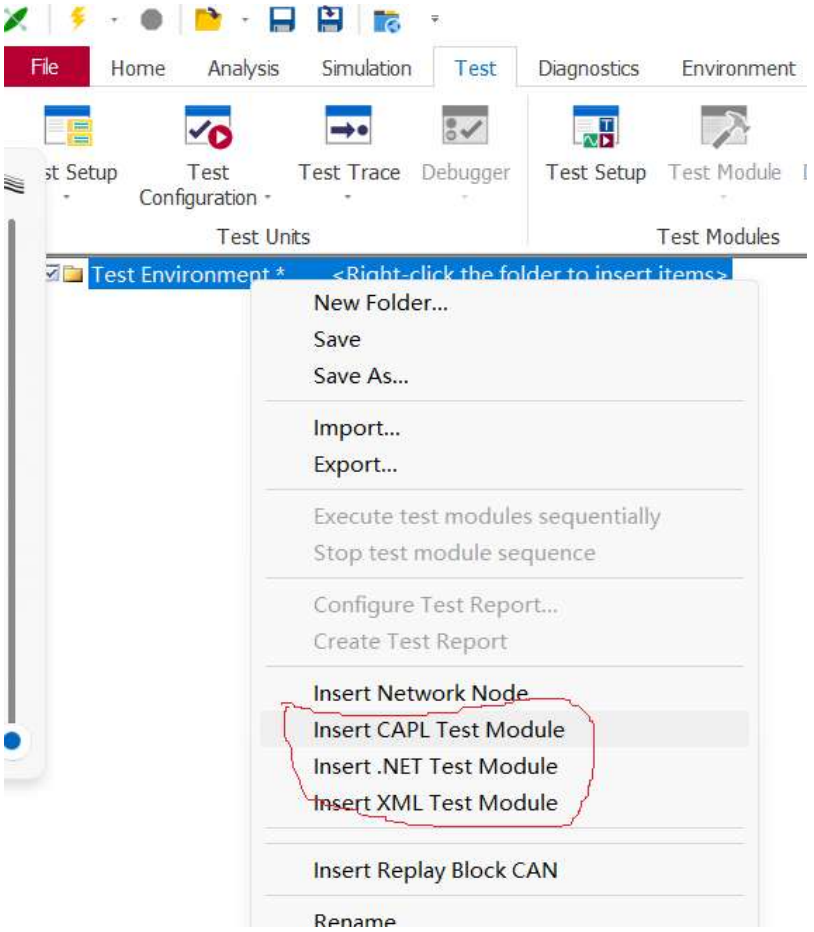
- 自动化测试是CANOE重要的功能,是实行HIL测试的重要工具。
- 我们点击TEST选项卡进入自动化测试开发界面
- 点击TEST SETUP进入脚本开发界面



常用的canoe自动化开发脚本

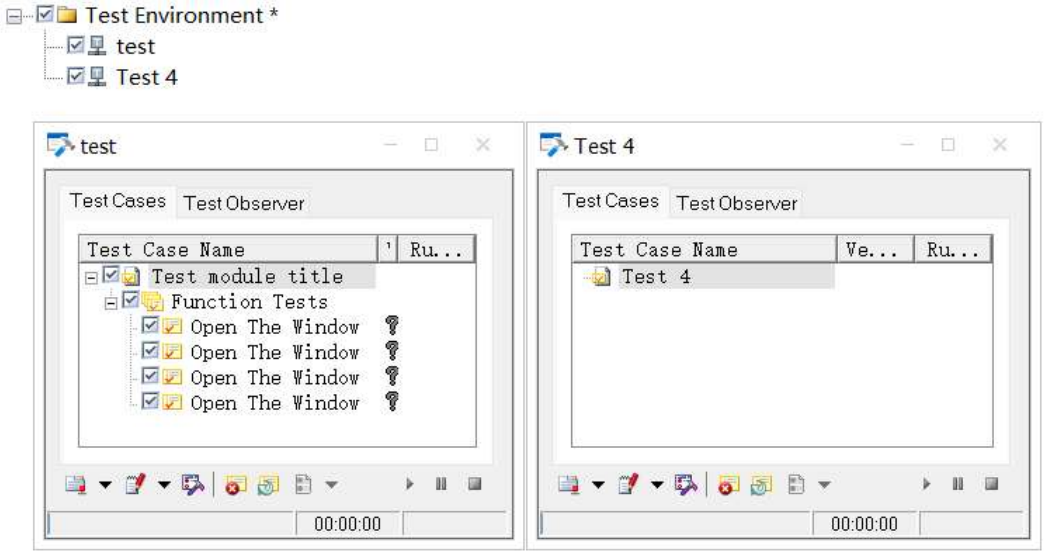
- 进入自动化界面创建自动化测试环境
- 自动化开发使用的脚本有：
  1. CAPL
  2. XML
  3. .NET





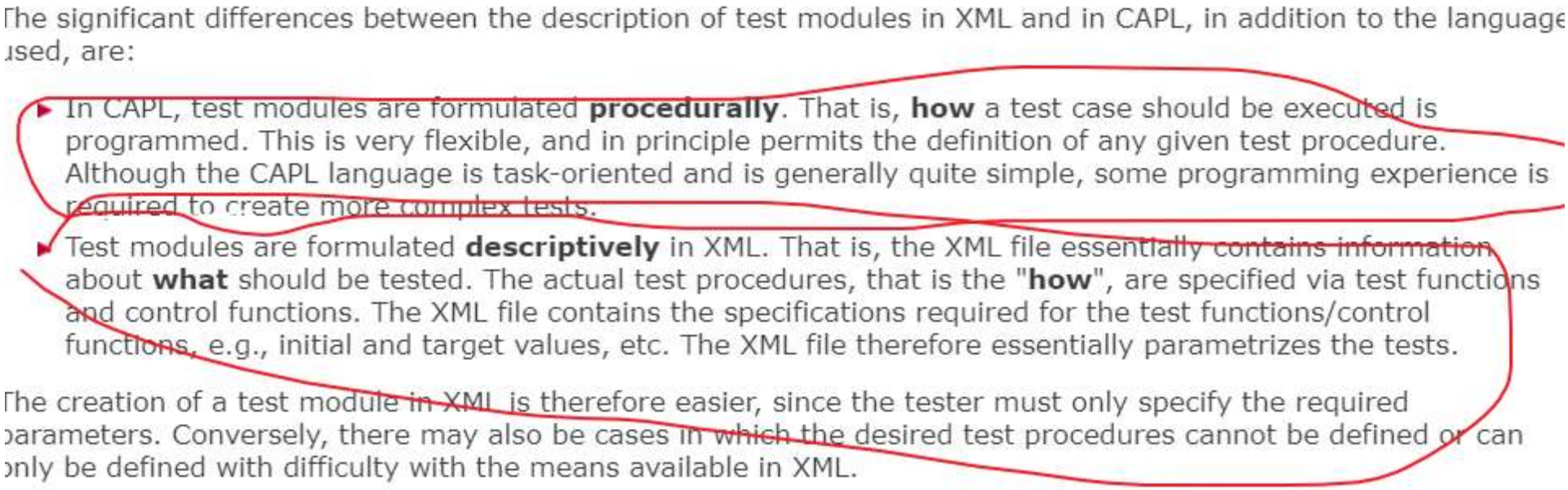
■为什么使用XML开发脚本

- CAPL是canoe开发脚本的主要语言，如果自动化开发使用CAPL在执行中不能自由选择执行的test case
- Xml开发自动化脚本中可以自由选择需要执行的 test case。如图所示同样功能脚本，执行界面是不同的。



■为什么使用XML开发脚本

- 官方文档中也承认了在复杂场合使用CAPL一些弊端。
- XML能实现流程管控和驱动代码之间的隔离，符合开闭原则。
- XML是通用数据格式文件，可以在不同语言下使用。



The significant differences between the description of test modules in XML and in CAPL, in addition to the language used, are:

- In CAPL, test modules are formulated **procedurally**. That is, **how** a test case should be executed is programmed. This is very flexible, and in principle permits the definition of any given test procedure. Although the CAPL language is task-oriented and is generally quite simple, some programming experience is required to create more complex tests.
- Test modules are formulated **descriptively** in XML. That is, the XML file essentially contains information about **what** should be tested. The actual test procedures, that is the "**how**", are specified via test functions and control functions. The XML file contains the specifications required for the test functions/control functions, e.g., initial and target values, etc. The XML file therefore essentially parametrizes the tests.

The creation of a test module in XML is therefore easier, since the tester must only specify the required parameters. Conversely, there may also be cases in which the desired test procedures cannot be defined or can only be defined with difficulty with the means available in XML.

■使用XML开发脚本

- XML文件简介：
  1. 可扩展标记语言 (Extensible Markup Language, XML)，标准通用标记语言的子集，可以用来标记数据、定义数据类型，是一种允许用户对自己的标记语言进行定义的源语言.
  2. XML可用于交换数据。基于XML可以在不兼容的系统之间交换数据，计算机系统和数据库系统所存储的数据有多种形式，对于开发者来说，最耗时间的工作就是在遍布网络的系统之间交换数据。把数据转换为XML格式存储将大大减少交换数据时的复杂性，还可以使这些数据能被不同的程序读取。
  3. CANOE的XML脚本遵守XML基本规则

■使用XML开发脚本

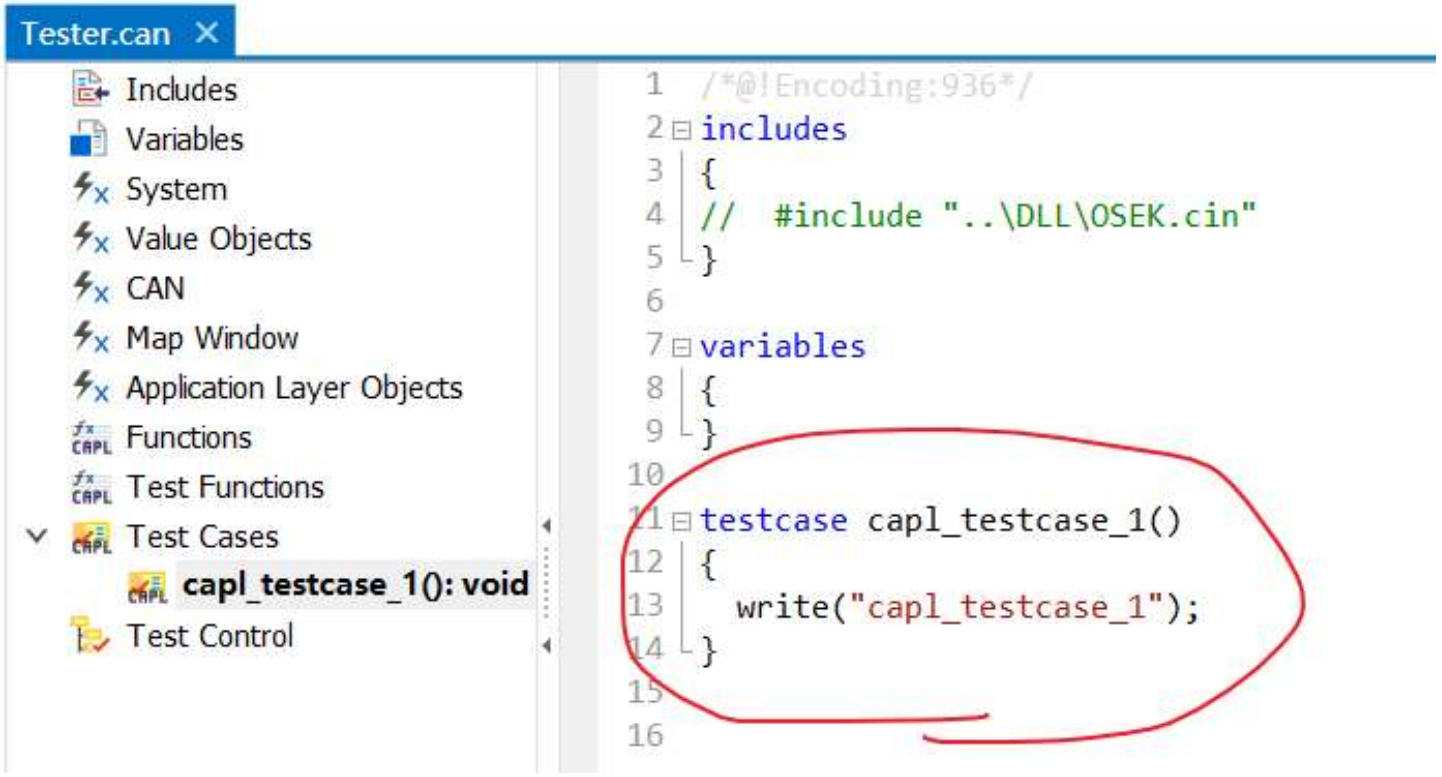
- XML脚本基本框架

1. 脚本只有一个根元素：testmodule。这个就是对应CANOE的自动化脚本开发环境。
2. 脚本可以定义多个二级元素：testgroup。同时也可以定义多个三级元素：testcase
3. 如下图范例定义两个testgroup,第一个testgroup有两个testcase
4. 第一个testcase调用CAPL脚本中的testcase::capl_testcase_1

```
1  <testmodule title="Test module title" version="Test module
   version">
2      <testgroup ident="1.0" title="test case1" >
3          <capltestcase name="capl_testcase_1" >
4          </capltestcase>
5          <capltestcase name="capl_testcase_1.1" >
6          </capltestcase>
7      </testgroup>
8      <testgroup ident="2.0" title="test case2" >
9          <capltestcase name="capl_testcase_2" >
10         </capltestcase>
11     </testgroup>
12 </testmodule>
```

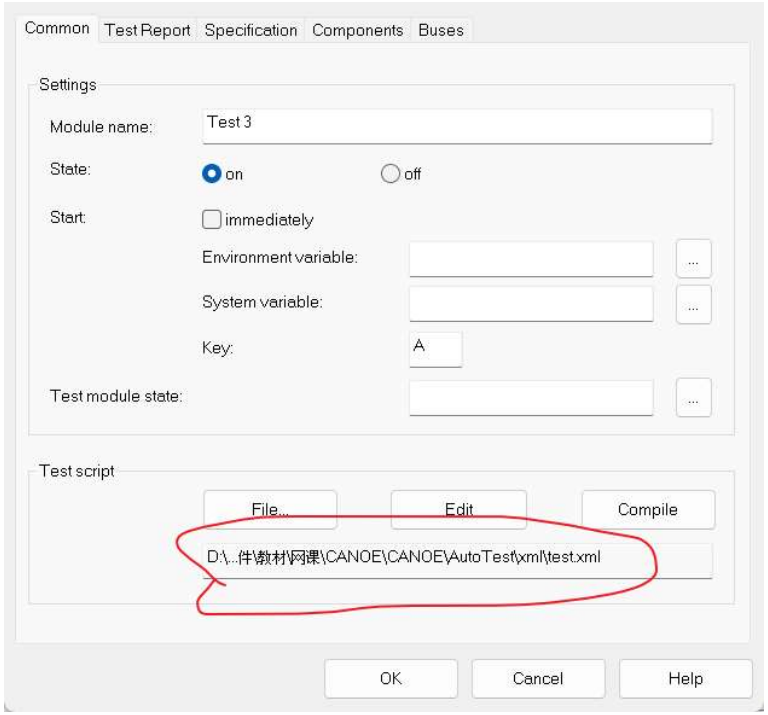
■使用XML开发脚本

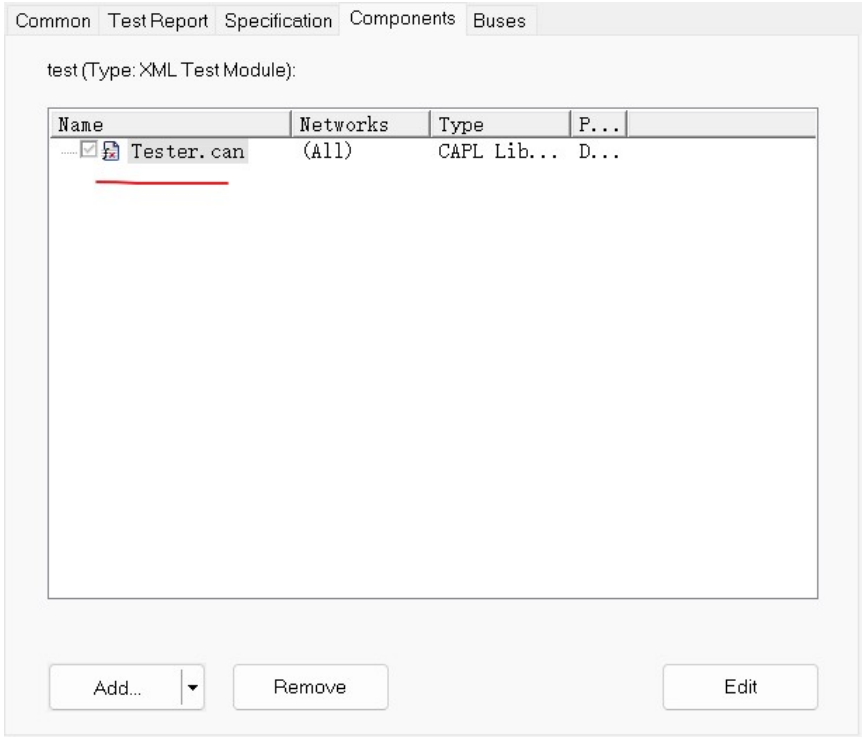
- CAPL脚本基本框架
  1. CAPL必须使用testcase定义，名称和XML脚本一致
  2. CAPL脚本中不能有main函数，避免重复调用
  3. 范例中只显示名称，真正项目应用中可以把变量放入XML导入CAPL脚本



■使用XML开发脚本

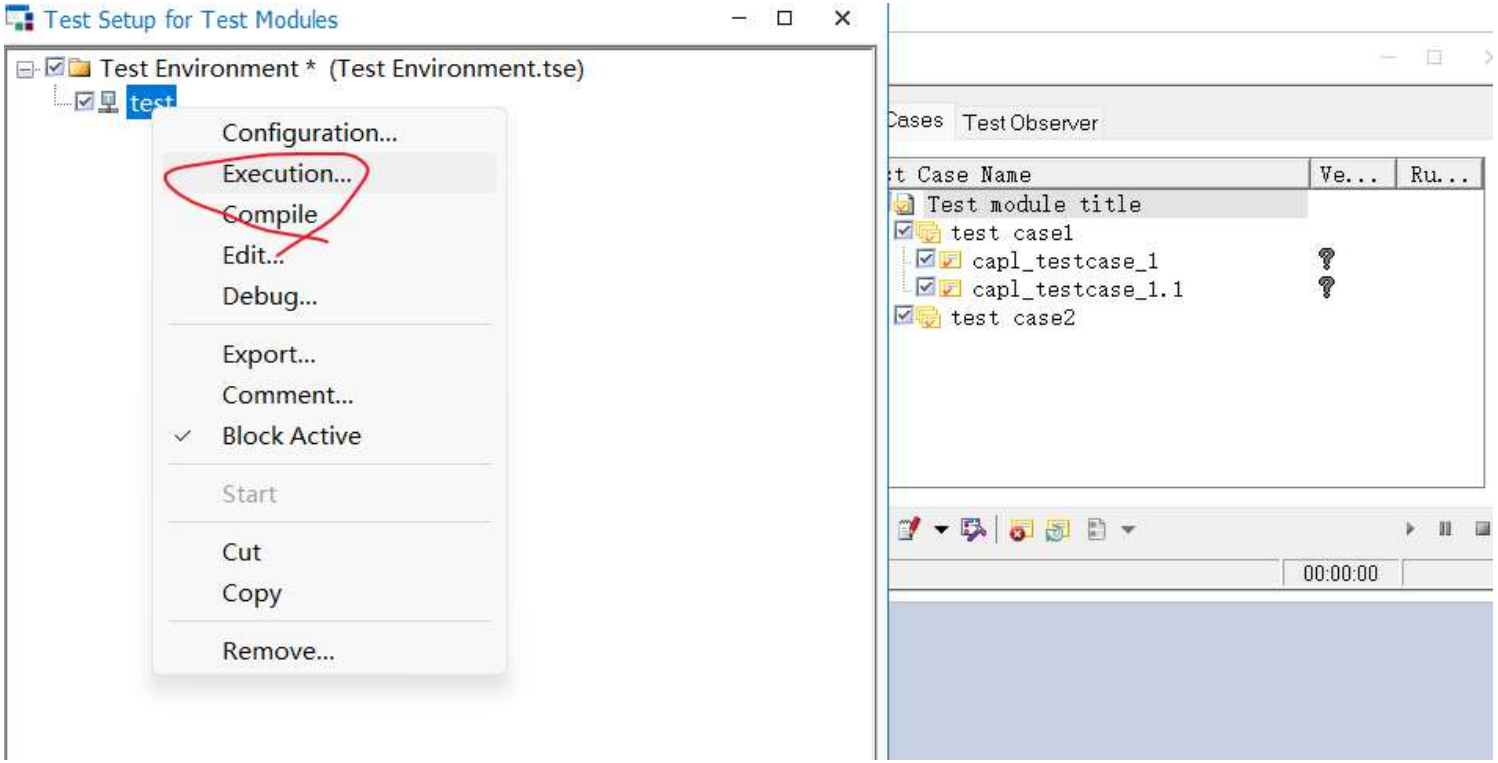
- XML脚本基本框架基本逻辑
  1. XML脚本中没有逻辑控制代码，所以要在CAPL中实现。
  2. 设置和仿真环境下的Node设置一样，只是CAPL文件放入Component选项卡中
  3. CAPL就好像DLL一样被XML调用





■使用XML开发脚本

- XML脚本基本框架
  1. Compile对应的XML脚本，如果有错误则再write窗口弹出错误信息。
  2. 再点击execute就会看到执行脚本上出现设置的case
  3. 再case左边有了勾选项，如果使用CAPL则没有，这也是XML能实现开闭原则的结果



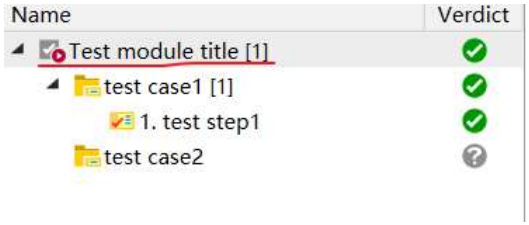
■使用XML开发脚本

- 上述脚本只是演示XML调用capl，在现实项目应用中需要增加一些功能：
  1. XML和CAPL之间的数据接口
  2. 测试报告中增加的描述信息
  3. CAPL脚本的逻辑功能
  4. XML增加注解信息

■使用XML开发脚本

- XML和CAPL之间的数据接口(TestModule)
  1. Title
  2. Version
  3. Description(option)
- 其中title会显示在报告中

```
<?xml version="1.0" encoding="iso-8859-1" standalone="yes"?>
<testmodule title="Test module title"  version="1.0">
    <description>Test module description demo string</description>
    <testgroup ident="1.0" title="test case1" >
```



■使用XML开发脚本

- XML和CAPL之间的数据接口(Testgroup)
  1. ident
  2. Title
  3. version(option)
  4. description(option)

```
<testgroup ident="1.0" title="test case1" version="1.0" >
    <description>Test group description demo string</description>
```

- 其中title会显示在报告中

Main Part

Time Stamp	Ident	Title
5.060000		1. **test step1**

■使用XML开发脚本

- XML和CAPL之间的数据接口(Testcase)
  1. 调用脚本类型：capltestcase；testcase；nettestcase
  2. Name
  3. Title
  4. 注入脚本的数据：Caplparm，netparm，cansignal

```
<capltestcase name="capl_testcase_1" title="test step1" Ident="1.1">
    <caplparam name="a" type="int">100</caplparam>
</capltestcase>
```

■使用XML开发脚本

- XML和CAPL之间的数据接口
  1. XML中使用Caplpara设置接口数据；CAPL使用testcase接受数据
  2. 设置name,type。

```
<caplparam name="a" type="int">100</caplparam>
```

```
testcase capl_testcase_1(int a)
{
  if(a==10)
  {
    teststeppass("value=10");
  }else
  {
    teststepfail("value!=10");
  }
  write("%d=a",a);
  write("capl_testcase_1");
}
```

■使用XML开发脚本

- XML和CAPL之间的数据接口

谢谢观看

Chart 20