我所增加和改变的都在Apps\CoAP.Tester目录下

standard.xml中存储了标准测试用例。

以root节点作为根节点，每个根节点下有多个testCase节点，一个testCase表示一个测试用例。testCase中包含testCaseItem，testCaseItem中包含testCaseRequest和testCaseResponse表示测试序列。具体子项表示不同的含义。

result.xsl是一个定义文件，定义了执行结果导出的格式，执行结果把XML内容渲染为HTML，以方便浏览器打开查看。

RowItem.cs文件中定义了RowItem类、StepItem类、和TestCaseItem类。

TestCaseItem类与Standard.xml中的testCaseItem节点相对应，存储了名称、组名、测试是否通过、一组StepItem和操作界面的委托。

StepItem类包含了名称、是否通过、测试类型（request还是response）

程序中包含静态类TestCaseXMLToObject，该类主要定义了一些对XML文件进行IO操作的函数。读取与standard.xml格式一致的测试用例存储文件。

程序中包含静态类TestCaseResultIO，该类主要定义了生成测试用例运行结果文件（html格式）的一些函数。

定义了基类TestCaseBase，并且TestCaseResponse类和TestCaseRequest类均继承自TestCaseBase。

定义了类TestCase，该类与standard.xml文件中的TestCase节点相对应。该类包含了一组TestCaseBase和Group，Identifier，Objective，Configuration等，还有比较函数Compare。

RequestBuildStepForm.cs类和ResponseCheckStepForm.cs类分别表示请求对话框和响应对话框。

RequestBuilders.cs文件中包含RequestBuilders(静态类)、MethodBuilder、MessageTypeBuilder、TokenBuilder、OptionBuilder、PayloadBuilder和GroupBuilder。其中RequestBuilders类包含了一些创建上述类的对象的函数。这些对象旨在包装请求。

Checks.cs定义了所有的检测类和函数，检测接收的数据和预期的数据是否一致。

软件执行流程如下：

LoadDefaultTestCase载入默认测试用例。

点击“运行用例”按钮执行RunTestCase操作，打开新的线程执行测试用例。

程序中用到了多线程，在另一个线程中需要控制主线程中的控件的显示，用到了委托。