第33卷 第13期 计 算 机 工 程 2007年7月
Vol.33 No.13 Computer Engineering July 2007
·软件技术与数据库· 文章编号：1000—3428(2007)13—0096—03 文献标识码：A 中图分类号：TP311.52

基于 XML 的 API 自动化测试工具设计与实现

刘慕涛[1]，张 磊[2]，王 艳[3]，周晓中[1]，张红雷[1]，左 芸[2]
（1. 总参0526工程办公室，北京 100081；2. 华东计算技术研究所，上海 200233；3. 空军装备研究院总体论证研究所，北京 100076）

摘 要：在研究应用编程接口(API)测试特点的基础上，运用软件测试技术与方法对API进行了正确性测试，设计了API自动化测试框架，采用XML语言进行API自动化测试工具的设计和实现方案。使用该工具可在API信息提取、测试用例生成及测试用例执行、测试报告生成等方面实现自动化。该工具主要用于验证第三方提供的API，由于单元测试中较多地使用了API测试，因此也可运用于软件的单元测试。
关键词：应用编程接口测试；软件测试驱动；软件测试自动化

Design and Implementation of Automatic API Test Tool Based on XML

LIU Mutao[1], ZHANG Lei[2], WANG Yan[3], ZHOU Xiaozhong[1], ZHANG Honglei[1], ZUO Yun[2]
(1. 0526 Project Office, General Staff, Beijing 100081; 2. East China Research Institute of Computer Technology, Shanghai 200233;
3. Research Institute of General Demonstration and Evolvement of Airforce Equipment, Beijing 100076)

【Abstract】 Based on the research of API test, this paper uses software test technology to do API validity test, and designs the frame of automatic API test. It also gives the design and implementation scheme of automatic API test tool by using XML language and automation of API information pick-up, test case generation and test case implementation comes true with it. This tool is mainly used to verify the functions of third-party APIs and is adapted to software unit tests because of frequent use of API test in unit tests.
【Key words】 application program interface (API) test; drive of software test; automatization of software test

软件测试是软件生命周期的一个重要阶段。软件测试技术，特别是软件测试自动化技术是当前国际软件界最有争议、亟待发展的技术。自动化测试就是通过自动化测试工具或其他手段，按照测试工程师的预定计划进行自动测试，目的是减轻手工测试的劳动量，提高软件质量和软件测试效率[1,3]。

随着软件工程和软件开发技术的不断发展，在软件开发过程中大量采用分层的软件体系结构。在分层的软件体系中，下层为上层提供应用编程接口(application program interface, API)作为接口开发者与接口使用者之间的合约。API作为软件体系中的基本单位，其质量直接关系到最终的软件质量，因此，对API进行测试非常重要。但API种类繁多，对每个API进行测试的工作量非常大，因此，研究API的测试方法及如何对其进行自动化测试有较强的实用价值和学术意义。

1 API测试及其自动化

API测试是运用软件测试技术与方法对API进行的正确性测试，包括逻辑正确性、功能正确性等。测试一般由下列过程组成：(1)建立初始条件；(2)用符合要求的参数调用API；(3)结果分析。

通过对API测试过程的研究，可以发现其中有很多重复性很强的工作，极为适合于自动化。通常把高度采用自动化的软件测试称为自动化测试。对于人机交互界面的测试，因为经常用到图形界面输出，不容易实现自动化测试，所以一般把图形界面的输出独立出来，单独建立测试；API测试一般不涉及图形界面的显示，并且可以准确地预计输出结果，所以很容易进行自动化测试。自动化测试一般包含如下基本过程：

(1)测试设计：设计测试用例，搭建测试环境等；

(2)脚本生成：根据测试设计生成需要运行的测试脚本，对于API自动化测试来说，测试脚本就是能够驱动被测API运行的一段代码；

(3)脚本运行：脚本运行需要将被测API的驱动代码编译成可执行文件来运行；

(4)结果比较：主要是分析脚本运行得到的结果是否与预期的相符，以此决定测试是否通过；

(5)测试报告生成：对测试结果进行分类整理，生成相关的测试报告，对于不能通过的测试结果进行分析、分类、记录和通报，让相关测试人员和开发人员了解测试结果。

2 API自动化测试框架的设计和实现

本文依据上述API自动化测试过程，设计了API自动化测试框架，如图1所示。该框架主要由API信息解析及管理模块、测试脚本生成模块、用例运行模块、报告生成模块以及测试管理环境接口模块构成。各个子模块实现的功能如下：

(1)API信息解析及管理模块：负责被测API信息的提取及管理。首先对待测API的头文件与源文件进行静态分析，得到API的相关信息，然后将所有的API信息以一定的规则组织起来生成API信息XML文件。API信息使测试脚本生

作者简介：刘慕涛(19—)，男，工程师，主研方向：软件工程与软件测试技术；张 磊，工程师、硕士；王 艳，工程师；周晓中，高级工程师、博士研究生；张红雷，高级工程师、硕士；左 芸，硕士
收稿日期：2007-05-28 **E-mail：**hangtianlmt@163.com

成模块知道如何生成测试用例的描述信息，进而生成 API 驱动代码，所以，完整、全面的 API 信息是 API 测试能够自动化进行的关键。

(2)用例生成模块：把 API 信息解析及管理模块生成的 API 信息文件生成测试用例描述 XML 文件。

(3)用例驱动脚本生成模块：把 API 信息解析及管理模块生成的 API 信息文件和测试用例描述 XML 文件提供的信息生成用例驱动脚本。

(4)用例运行模块：为用例驱动程序的自动执行提供运行框架，并生成执行记录 XML 文件。

(5)报告生成模块：使用用例运行模块生成的执行记录 XML 文件来生成符合用户标准的测试报告文件。在实际使用过程中，该模块可以根据不同的用户要求进行扩展。

(6)测试管理环境接口模块：提供与上层测试管理环境的接口，测试管理环境可以通过该接口向测试工具下达任务，指导测试用例的生成。测试工具执行完毕后向管理环境返回用例执行结果，并能够将测试过程中生成的各种脚本及文件通过测试管理环境入库，方便进行回归测试。

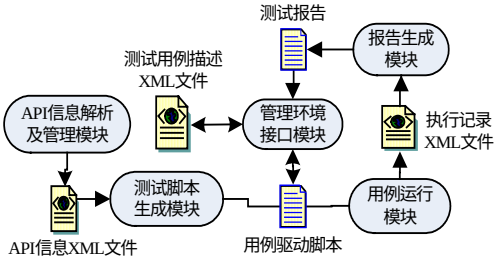

图 1　API 自动化测试框架

由于XML语言具有简洁、易扩展等优点，特别适用于结构化数据的描述，因此各个模块之间使用XML文件作为信息交换的载体，具有松散耦合的结构。这种松散耦合的结构方便工具进行扩展和升级，另外，XML语言的平台无关性也方便工具进行分布式扩展[2,4]。

3　API 自动化测试工具的实际运用

从上面的描述中可以看出，各类 XML 文件及用例脚本文件在工具中占有核心地位，因此，定义好各个 XML 文件的内容是用好该工具的关键。下面结合一个 C 语言的例子来说明该工具在实际运行中生成的各种文件及测试结果。

被测源代码文件如下：

```
int add(int i,int j)
{ return i+j;        }
```

3.1　API 信息 XML 文件

依据 API 测试过程设计的 API 信息文件包含如下内容：

(1)建立初始环境所需信息，主要包括对象的初始化信息（面向对象语言）、使用的外部资源信息、依赖的其他 API 信息等；

(2)调用（驱动）API 所需信息，主要包括 API 调用方式信息、参数信息、单元测试打桩所需信息等；

(3)输出和分析测试结果所需信息，主要包括返回值、指针参数或引用参数、抛出的异常、全局变量、文件和外部设备等信息。

下面是使用 API 信息解析及管理模块对例子进行静态分析而生成的 API 信息 XML 文件。

```
<?xml version="1.0" encoding="gb2312"?>
 <sourceFile  fileName="D:\a.c"  size="42"  language="c"
modTime="2007-4-4">
        <functionList>
         <function startLine="1" endLine="3" isImplement="true">
           <returnType>int</returnType>
           <funcName>add</funcName>
           <argList>
             <argument isAtomic="false">
               <argType>int</argType>
               <argName>i</argName>
               <argValue></argValue>
               <argOrder>1</argOrder>
             </argument>
             <argument isAtomic="false">
               <argType>int</argType>
               <argName>j</argName>
               <argValue></argValue>
               <argOrder>2</argOrder>
             </argument>
           </argList>
         </function>
        </functionList>
      </sourceFile>
    </sourceFiles>
```

3.2　测试用例描述 XML 文件

只有对测试用例进行正确、详细、周全的描述才有可能实现有意义的测试。在一次测试活动中可能要将多个测试用例组织在一起运行，这时候多个测试用例就会有公共的描述信息，每个测试用例又会有自己私有的描述信息，因此，将测试用例的描述信息分为公共信息和私有信息 2 种。

测试用例的公共描述信息包括如下内容：

(1)描述信息：用来指明这一组相关的测试用例的测试目的，使测试人员及其他相关人员能清楚地了解测试的意义，方便用例的查找与定位。

(2)初始化操作：描述在对一组用例进行测试之前要进行的准备工作，包括变量的赋值、环境的配置等工作。

(3)清理操作：描述在对一组用例进行测试之后进行的善后工作，其目的是使系统恢复到测试之前的状态，包括资源的释放等工作。

(4)公共资源：描述这一组测试用例所需要的共同信息，比如全局变量信息、桩类、桩函数等。

(5)用例组织信息：描述各个测试用例的私有描述信息所在位置，目的是使通过访问用例的公共描述信息能够方便地找到各个用例的私有描述信息。

测试用例的私有描述信息包括如下内容：

(1)描述信息：用来指明该测试用例的测试目的，使测试人员及其他相关人员能清楚地了解测试的意义，方便用例的查找与定位；

(2)名称：描述该测试用例的标识；

(3)序号：描述该测试用例在一组测试用例中的位置；

(4)参数信息：描述该用例所测 API 的每个参数的详细信息，包括各个参数的名字、输入值、预期值、在参数表中的位置。对于依据边界值和等价类划分自动生成测试数据的情况，还要指定参数值的上下限和参数值递增的步长；

(5)初始化操作：描述在被测 API 运行之前要进行的准备工作，包括变量的赋值和相关环境的配置等操作；

(6)被测 API 运行前的输出操作：用来输出被测 API 运行

前的一些信息，包括全局变量的值、参数的值等，用来跟被测 API 运行后的值进行对比；

(7)被测 API 运行后的输出操作：用来输出被测 API 运行后的一些信息，包括全局变量的值、参数的值、API 返回值等，用来跟被测 API 运行前的值进行对比，以及跟用户的预期值进行对比等。

将例子中的 API 函数第 1 个参数值赋为 2，第 2 个参数值赋为 3，预期结果为 5，生成的测试用例描述文件为

```
<?xml version='1.0' encoding="GB2312" ?>
<testCase isActive = "true" targetSourceFile = "D:\a.c" targetMethod = "add" >
<path>testsuite/seq1/1/tcxml/1-1.xml</path>
<description></description>
<name>1-1</name>
<order>1</order>
<parameter>
<name>i</name>
<value>2</value>
<order>1</order>
</parameter>
<parameter>
<name>j</name>
<value>3</value>
<order>2</order>
</parameter>
<expReturn>5</expReturn>
</testCase>
```

3.3 执行记录 XML 文件

执行测试用例的最终目的是判定用例所验证的功能是否正确，在自动化测试中，充分翔实的用例执行记录是判定用例是否通过的唯一依据。在用例的执行记录中应该包括时间信息、用例标识信息、变量运行值信息、判定结果信息等。

运行例子实际生成的执行记录 XML 文件如下：

```
<?xml version="1.0" encoding="gb2312" ?>
<result>
<timeStamp>
2007-04-04 10:46:34
</timeStamp>
<targetApi>
<tcName>1-1</tcName>
<apiName>int add(int:2,int:3)</apiName>
<beforeInvoke>
<parameter>
</parameter>
<global>
</global>
<customized>
</customized>
</beforeInvoke>
<afterInvoke>
<return>
<variant name="__api_var_return" value="5" expValue="5" notes="__api_var_return==5" assertion="passed" isAtomic="true"/>
</return>
</afterInvoke>
<assertion>passed</assertion>
</targetApi>
</result>
```

从该文件中可以看出，被测 API 实际输入的结果为 5，与预期的完全一致，因此，可以判定该用例运行通过。

4 总结

本文给出了一种 API 自动化测试工具的设计和实现方案，使用该工具可在 API 信息提取、测试用例生成及测试用例执行、测试报告生成等方面实现很大程度的自动化。目前该测试工具已开发完成，通过实际的使用证明，该工具能够对 API 进行高效准确的测试，极大地提高了测试效率，对软件质量的提高也有很大帮助。

参考文献

1 Mosley D J, Posey B A. 软件测试自动化[M]. 邓 波, 黄丽娟, 译. 北京: 机械工业出版社, 2003.

2 Holzner S. XML 完全探索[M]. 师夷工作室, 译. 北京: 中国青年出版社, 2001.

3 Dustin E, Rashka J, Paul J. 软件自动化测试: 引入、管理与实施[M]. 于秀山, 胡兢玉, 译. 北京: 电子工业出版社, 2003.

4 Wegener, Joachim, Baresel, et al. Evolutionary Test Environment for Automatic Structural Testing[J]. Information and Software Technology, 2001, 43(14): 841-854.

（上接第 92 页）

同样依据IEM计算方法，对区间矩阵 C_2 可求得 W_1=[0.137,0.157]，W_2=[0.448,0.503]，W_3=[0.311,0.340]。

对区间矩阵 C_3 则可求得 W_1=[0.560,0.612]，W_2=[0.285, 0.317]，W_3=[0.105,0.111]。

由上面准则层和方案层权重系数，可得层次总排序结果如表 8 所示。

表 8 层次总排序结果

层次 P	层次 C			层次 P 总排序
	C_1	C_2	C_3	
	0.25	0.50	0.25	
S_1	[0.562，0.60]	[0.197，0.238]	[0.179，0.212]	[0.350，0.382]
S_2	[0.137，0.157]	[0.448，0.503]	[0.311，0.340]	[0.345，0.391]
S_3	[0.560，0.612]	[0.285，0.317]	[0.105，0.111]	[0.227，0.251]

表 8 中最右边一列即为层次总排序结果，对其应用 1.5 节中所定义的区间排序规则进行排序可得出最终评价结果：[0.345,0.391] > [0.350,0.382]> [0.227,0.251]。即供应商S_1>供应商S_1>供应商S_3，供应商S_2为本次评价中的首选供应商。

4 结论

基于混合 AHP 方法的供应商评定模型的研究与其在企业中的实际应用，说明基于混合 AHP 法的供应商评价方法是一种有效的量化方法，可以克服应用标准 AHP 方法进行供应商评定时所产生的不确定性弊端，保障企业科学合理地进行供应商评价和选择。

参考文献

1 樊治平, 潘德惠. 不确定判断矩阵权重计算的一种实用方法[J]. 系统工程, 1996, 14(2).

2 杨忠海, 滕春贤. AHP 在高新技术企业供应商选择方面的应用研究[J]. 哈尔滨商业大学学报, 2002, 18(5).

3 许先云, 杨永清. 不确定 AHP 判断矩阵的一致性逼近与排序方法[J]. 系统工程理论与实践, 1998, 18(2).

4 许先云. 基于区间收益的风险决策方法[J]. 江南学院学报, 2000, 15(2).

5 朱铎辉, 吴志军, 张玉峰. 基于层次分析法的供应商评价模型的研究[J]. 计算机应用研究, 2004, 21(4).