

Software
Testing

软件测试丛书

软件自动化测试成功之道：

典型工具、脚本开发、测试框架和项目实战

Software Automation Testing Best Practice

陈能技 编著

- 覆盖自动化测试前期准备、测试框架搭建、自动化测试过程管理等项目测试全过程
- 重点介绍自动化测试的脚本编写技巧，标准GUI、第三方控件和自定义控件测试等各种自动化测试的实战技术，以及主流测试工具QTP、TestComplete的实战知识
- 贴近实战的案例，如自动化测试框架搭建、自动化冒烟测试、每日构建最佳实践，对象驱动、关键字驱动测试等
- 配书光盘中赠送200多分钟的教学视频和全部源程序
- 帮助读者快速掌握自动化测试方法、技术和技巧



人民邮电出版社
POSTS & TELECOM PRESS



200分钟视频
全部源程序



Software
Testing

软件测试丛书

软件自动化测试成功之道：

典型工具、脚本开发、测试框架和项目实战

Software Automation Testing Best
Practice

陈能技 编著

人民邮电出版社

北京

图书在版编目 (C I P) 数据

软件自动化测试成功之道：典型工具、脚本开发、
测试框架和项目实战 / 陈能技编著. — 北京：人民邮电
出版社，2010.4

ISBN 978-7-115-21603-8

I. ①软… II. ①陈… III. ①软件—测试 IV.
①TP311.5

中国版本图书馆CIP数据核字(2009)第195123号

内 容 提 要

自动化功能测试是目前测试领域的发展方向，自动化功能测试能有效地降低测试成本、提高测试效率和测试覆盖面。本书结合项目实践全过程，对自动化测试前期准备阶段、测试框架搭建方法、自动化测试的过程管理进行了详细讲解，并介绍了各种自动化测试技巧，包括 GUI 控件的测试技巧、自动化脚本编写技巧等，以便让读者更好地学以致用。

本书分成两篇，第一篇是项目实践篇，主要介绍如何借助测试工具有效地开展功能自动化测试项目，包括自动化测试的计划、设计、脚本开发和维护的全过程管理。第二篇是实用技巧篇，主要介绍如何基于流行的自动化测试工具 QTP 设计测试框架，除介绍几个常用的框架外，还将重点介绍笔者设计的自动化测试框架 QTestWare。

本书适合自动化测试工程师、自动化测试项目管理者、项目经理、希望从事自动化测试工作的测试人员，以及希望了解自动化测试框架搭建技术的测试人员阅读。

软件自动化测试成功之道： 典型工具、脚本开发、测试框架和项目实践

-
- ◆ 编 著 陈能技
责任编辑 张 涛
 - ◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号
邮编 100061 电子函件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
三河市潮河印业有限公司印刷
 - ◆ 开本：800×1000 1/16
印张：26.25
字数：585 千字 2010 年 4 月第 1 版
印数：1—3 500 册 2010 年 4 月河北第 1 次印刷

ISBN 978-7-115-21603-8

定价：55.00 元（附光盘）

读者服务热线：(010)67132692 印装质量热线：(010)67129223
反盗版热线：(010)67171154

前 言

软件测试自动化是软件测试未来发展的重要方向，目前受到众多软件企业的关注，测试项目纷纷引入自动化的功能测试，期待通过自动化的功能测试达到提高测试效率、提高测试覆盖率的目的。但在企业引入自动化测试的过程中，也遇到了各种各样的问题，很多企业耗费巨资买入自动化测试工具，但由于缺乏合理的管理过程，缺少自动化测试实战技术的指导，导致测试工具在使用了一段时间后就被束之高阁，达不到应有的目的。

本书结合笔者在自动化测试项目管理方面的实战经验，为广大测试人员介绍成功实施自动化测试的完整过程，包括自动化测试开展的前期准备工作、测试框架搭建方法、自动化测试的过程管理，并且详细介绍了自动化测试过程中经常遇到的各种问题的解决方案，是自动化测试工程师的必备参考书，也是企业进行自动化测试的指南。

本书的内容安排

本书分为两篇，共 20 章。

第一篇是项目实践篇（第 1 章至第 18 章），主要介绍了自动化测试项目的前期准备工作；自动化测试工具的原理；如何借助测试工具有效地开展功能自动化测试项目，如自动化测试的计划、设计、脚本开发和维护的全过程管理；如何解决自动化测试项目中遇到的典型问题，如 GUI 控件识别等；如何提高自动化测试项目的可维护性；如何管理自动化测试项目。

第二篇是实用技巧篇（第 19 章和第 20 章），主要介绍了基于 QTP 自动化测试工具的几个扩展自动化测试框架，包括 SAFFRON、FrameworkManager 等。除了介绍几个常用的框架外，还重点介绍了笔者自行设计的自动化测试框架 QTestWare 的思想和实现过程。

本书的特点

本书的特点主要体现在以下几个方面：

- 本书的编排采用循序渐进的方式，适合初级、中级读者逐步掌握软件测试的基本方法，软件自动化测试设计和管理的精髓；
- 本书在介绍各种测试方法和技术时，结合了许多浅显易懂的例子，方便读者自己进行实践和演练，在介绍软件测试脚本开发时更是提供了丰富完整的开发示例代码，读者可直接使用，或者根据自己的实际情况进行调整；
- 不拘泥于工具，本书所介绍的自动化测试设计思想和方法并不局限于某个测试工具；
- 实战例子丰富，全文贯穿了各种实战技术和技巧，如测试脚本开发、GUI 控件测试等，以方便测试工程师参考，并把实例应用到自己的自动化测试项目中；
- 实践性强，本书融合了作者在自动化测试项目管理方面的丰富经验，为测试人员答疑解惑，帮助广大测试人员解决自动化测试项目中遇到的各种实际问题。

视频和源代码

为了方便读者阅读和使用本书，随书配套光盘中特意提供了 200 多分钟的教学视频，并提供完整源代码。

适合阅读本书的读者

- 自动化测试工程师；
- 自动化测试项目管理者；
- 项目经理；
- 希望从事自动化测试工作的测试人员；
- 希望了解自动化测试框架搭建技术的测试人员。

本书由陈能技组织编写，参与编写和代码测试的人员有傅奎、赵杨、王津涛、梁洋洋、韦一平、孟庆昌、田瑜辉、张伟、刘庆治等。

由于编写时间仓促，加之水平有限，书中难免存在不妥和错误之处，恳请广大读者批评指正，联系邮箱为：zhangtao@ptpress.com.cn。

编者

目 录

第一篇 项目实践篇

第 1 章 自动化测试的前期准备工作 1	4.2 第三方控件的自动化测试与配置管理的关系..... 44
1.1 合适的项目实施自动化测试..... 2	4.3 C1Schedule 控件的测试方法..... 45
1.2 合适的测试用例实现自动化..... 2	4.4 如何在 C1TrueDBGrid 控件中定位行..... 46
1.3 对控件的熟悉程度与自动化测试成功实施之间的关系..... 5	4.5 C1TrueDBGrid 控件——分组显示..... 47
1.4 自动化测试计划..... 5	4.6 C1TrueDBGrid 控件——输入数据..... 48
1.5 自动化测试脚本开发规范..... 9	4.7 C1FlexGrid 控件的测试方法..... 52
1.6 自动化测试计划文档..... 18	4.8 XtraNavBar 控件的测试方法..... 54
第 2 章 自动化测试工具 21	第 5 章 自定义控件的测试方法 58
2.1 自动化测试工具的基本原理..... 22	5.1 NavigateBar 控件的测试方法..... 59
2.2 基于 GUI 对象识别的测试工具原理..... 22	5.2 FlexGridTree 控件的测试方法..... 60
2.3 “测试对象”与“运行时对象”..... 24	5.3 ImageListBox 控件的测试方法..... 66
2.4 DOM 与 IE Developer ToolBar..... 25	5.4 TSSTrueDBGrid 控件的测试方法..... 67
2.5 基于浏览器和 DOM 对象模型的功能自动化测试工具..... 28	第 6 章 自动化代码框架的搭建 71
第 3 章 标准控件的特殊问题的处理方法 32	6.1 模块化框架..... 72
3.1 识别控件的技巧..... 33	6.2 函数库结构框架..... 74
3.2 带 CheckBox 的 TreeView 控件的处理方法..... 33	6.3 关键字驱动框架..... 77
3.3 DataGridView 控件的处理方法..... 36	6.4 数据驱动框架..... 78
3.4 CheckedListBox 控件的处理方法..... 37	6.5 混合型框架..... 81
3.5 ToolStrip 控件的问题..... 39	6.6 编写框架范例..... 81
第 4 章 第三方控件的测试方法 42	6.7 是否使用多个项目..... 82
4.1 第三方控件的选择..... 43	第 7 章 典型公共用例 83
	7.1 登录模块..... 84

7.2 菜单选择.....	85	第 12 章 非预期窗口的处理.....	160
7.3 读入测试数据.....	86	12.1 非预期窗口的默认处理方法.....	161
7.4 根据功能模块打开菜单.....	89	12.2 非预期窗口的自定义处理方法.....	162
7.5 启动被测试应用程序.....	90	第 13 章 自动错误预防 (AEP) 机制的建立.....	167
7.6 退出被测试应用程序.....	91	13.1 AEP 的概念.....	168
7.7 配置测试数据库的连接串 (选择测试数据库).....	91	13.2 与手工测试、缺陷列表结合.....	168
7.8 “白+黑”——检查数据库的数据.....	97	13.3 与缺陷跟踪管理系统的整合.....	172
第 8 章 自动化冒烟测试.....	105	第 14 章 自动化测试模式.....	173
8.1 最简单级别的冒烟测试.....	106	14.1 ToolBar 按钮状态设置的检查.....	174
8.2 二级冒烟测试.....	107	14.2 查找功能的测试.....	181
8.3 Flight 程序的二级冒烟测试.....	109	14.3 提示信息的检查.....	188
第 9 章 与每日构建结合.....	117	14.4 消息框的处理.....	199
9.1 测试工具的命令行运行方式.....	118	14.5 关联数据的删除检查.....	201
9.2 通过 COM 访问和控制测试工具.....	121	14.6 文件上传下载过程的测试.....	206
9.3 与 Automated Build Studio 结合.....	124	第 15 章 手工测试用例的自动化实现.....	212
9.4 与 FinalBuilder 的结合.....	128	15.1 与测试用例库结合.....	213
9.5 导出测试脚本的运行日志.....	130	15.2 手工测试用例自动化实现的技巧.....	213
9.6 发送测试日志.....	132	第 16 章 UI 自动化测试.....	219
第 10 章 对象驱动测试 (ODT) 方法.....	138	16.1 自动化的用户体验测试.....	220
10.1 ODT 编辑.....	139	16.2 检查默认焦点位置.....	220
10.2 关联测试脚本中的函数.....	139	16.3 检查 Tab 顺序.....	223
10.3 类的方法.....	141	16.4 对检查 Tab 顺序的脚本进行完善.....	224
10.4 测试数据与类的关联.....	143	16.5 第二种检查 Tab 顺序的方法.....	228
10.5 测试数据与类的方法关联.....	144	16.6 检查字体设置.....	231
10.6 ODT 小结.....	148	16.7 检查界面元素的尺寸.....	234
第 11 章 关键字驱动测试框架.....	149	16.8 窗口标题根据菜单名称改变的问题.....	236
11.1 什么是关键字驱动.....	150	16.9 界面性能检查.....	238
11.2 在 TestComplete 中实现关键字驱动测试.....	150		
11.3 第二种方法的尝试.....	153		

第 17 章 自动化测试项目实践技巧245	数据库选择的问题.....265
17.1 数据库连接配置问题.....246	17.11 Log 的问题.....266
17.2 录制脚本的修改.....250	17.12 常用脚本函数的封装—— 字符串操作.....272
17.3 自动化测试脚本编写模式.....250	17.13 文件操作.....277
17.4 测试脚本编写原则.....253	第 18 章 自动化测试的管理282
17.5 CLR Bridge 的问题.....254	18.1 测试脚本的源代码控制规范.....283
17.6 Keys 还是 wText.....255	18.2 自动化测试项目中的交流.....285
17.7 中文命名的控件问题.....256	18.3 自动化测试工具的版本升级 问题.....288
17.8 回放脚本失败, 慎防 AUT 多进程问题.....259	18.4 自动化测试脚本的维护.....289
17.9 全局变量的问题.....261	18.5 自动化测试团队管理.....290
17.10 自动化测试脚本开发与	

第二篇 实用技巧篇

第 19 章 基于 QTP 的扩展自动化 测试框架291	19.2 FrameworkManager 框架.....310
19.1 SAFFRON 框架.....292	19.2.1 FrameworkManager 简介.....310
19.1.1 如何使用 SAFFRON.....292	19.2.2 FrameworkManager 的 特点.....311
19.1.2 SAFFRON 框架代码 剖析.....294	19.2.3 下载 Framework Manager.....312
19.1.3 导航到指定 URL.....294	19.2.4 FrameworkManager 框架的组成.....312
19.1.4 返回测试对象的描述.....295	19.2.5 对象树 Excel 文件.....312
19.1.5 启动浏览器.....295	19.2.6 clsTreeLevel 类.....314
19.1.6 给指定字符串前后加 双引号.....296	19.2.7 输入子类 and 随机值子类.....314
19.1.7 单击链接.....297	19.2.8 FrameworkManager 的 使用方法.....315
19.1.8 一个小例子.....299	19.3 QTP 中的 Process Guidance.....315
19.1.9 检查对象是否存在.....300	19.3.1 什么是过程指南.....316
19.1.10 在文本框输入字符串.....303	19.3.2 过程指南的作用.....318
19.1.11 读取文本框的字符串.....304	19.3.3 自定义过程指南.....320
19.1.12 选择列表中的一项.....306	19.3.4 过程指南深入探秘.....323
19.1.13 关闭浏览器.....307	19.4 业务流程测试框架.....328
19.1.14 对 SAFFRON 框架 进行扩展.....308	19.4.1 BPT 框架简介.....328

19.4.2	基于角色和工作流的 BPT 模型	328	20.2.8	DomainUpDown 控件	361
19.4.3	QTP 与 QC 的紧密整合	329	20.2.9	ListView 控件	362
19.4.4	版本控制和基线	330	20.2.10	MaskedTextBox 控件	363
第 20 章	QTestWare 框架的设计	332	20.2.11	MenuStrip 控件	364
20.1	QTestWare 框架介绍	333	20.2.12	PropertyGrid 控件	365
20.1.1	QTestWare 支持的环境	333	20.2.13	StatusStrip 控件	368
20.1.2	QTestWare 解决控件 录制问题	334	20.2.14	ToolStrip 控件	368
20.1.3	QTestWare 解决控件 识别问题	334	20.2.15	TrackBar 控件	370
20.1.4	QTestWare 让脚本更 简短	334	20.2.16	TreeView 控件	374
20.1.5	QTestWare 封装了高 可重用性的脚本	335	20.3	用 QTestWare 解决 Web 控件 问题	376
20.1.6	QTestWare 采用的技术 和原理	335	20.3.1	Browser 对象	376
20.1.7	QTestWare 专注解决的 框架问题	337	20.3.2	Page 对象	379
20.2	用 QTestWare 解决 .NET 控件 问题	338	20.3.3	WebTable 对象	381
20.2.1	CheckedListBox 控件	338	20.3.4	WebList 对象	382
20.2.2	ComponentOne 之 FlexGrid 控件	339	20.3.5	YUI 之 Calendar 控件	384
20.2.3	ComponentOne 之 TrueDBGrid 控件	341	20.3.6	YUI 之 DataTable 控件	388
20.2.4	ComponentOne 之 ToolBar 控件	342	20.3.7	YUI 之 Slider 控件	390
20.2.5	ComponentOne 之 MainMenu 控件	348	20.3.8	YUI 之 TreeView 控件	394
20.2.6	ContextMenuStrip 控件	357	20.4	QTestWare 框架通用函数	397
20.2.7	DataGridView 控件	358	20.4.1	CaptureBitmapAndSave ToLog 用于截屏并保存 到日志	397
			20.4.2	DeviceReplay 用于控制 鼠标和键盘	399
			20.4.3	GetToolTip 用于获取 ToolTip 上的文本	405
			20.4.4	MousePosition 用于获取 鼠标光标坐标	406
			20.4.5	EnterNode 和 ExitNode 用于创建层次化的测试 报告	407

第一篇

项目实践篇

Chapter

1

第1章

自动化测试的前期 准备工作

在项目启动阶段，就可以开始一些自动化测试准备工作了，包括如下几个方面：

- 编写自动化测试用例；
- 封装第三方控件、自定义控件的测试方法；
- 制订测试脚本编码规范。

1.1 选择合适的项目实施自动化测试

很多企业在实施自动化测试时，往往会进入一个误区，认为进度紧、测试资源不够的情况下，可以通过自动化测试来减轻测试人员手工测试的负担，从而更快地完成测试任务。

而事实上，自动化测试无论是对人还是对进度而言，都与普通的开发项目一样，需要投入足够的资源和时间进行自动化测试的计划、设计和脚本开发。

因此，挑选一个进度不紧测试人员相对充裕的测试项目来开展自动化测试，尤其是对于初次尝试自动化测试项目的组织而言成功率会高很多。

自动化测试只有在多次运行后，才能体现出自动化的优势，只有不断地运行自动化测试，才能有效预防缺陷、减轻测试人员手工回归测试的工作量。如果一个项目是短期的，并且是一次性的，则不适合开展自动化测试，因为这种项目得不到自动化测试的应有效果和价值体现。

1.2 选择合适的测试用例实现自动化

对于自动化测试团队而言，容易犯的一个典型的错误是：没有选择恰当的测试用例来实现自动化测试。

大部分自动化测试项目失败的原因主要归咎于被测试应用程序的快速变化、不恰当的测试用例、不可靠的框架以及脚本编程的问题。分析这些问题的根源，我们可以看到，自动化测试必须分阶段逐步开展，而不能局限在某个阶段完成自动化测试。因此，建议自动化测试从选择那些重要的、合适的测试用例开始，然后慢慢地扩展到其他方面。这样会带来较低的维护成本，能实现更重要的业务价值。

那么，如何选择合适的测试用例呢？

通常需要结合测试用例复杂度的评估来考虑选择的测试用例以及个数。首先把测试用例按一定的原则分为简单、中等、复杂3大类。然后从这3大类的测试用例中按一定的比例来选取需要实现自动化测试的用例。

测试用例的复杂度分组可以通过综合分析测试用例包含的测试步骤（操作步骤），以及测试用例所包含的检查点个数来判定。例如，可参考表 1.1 来分类。

表 1.1 测试用例复杂度分类

序 号	测试用例复杂度	操作（步骤）的个数	检查点的个数
1	简单	< 5	< 5
2	中等	5~15	5~10
3	复杂	15~25	10~15

从表 1.1 中可以看出。

(1) 如果测试用例中包含的测试步骤个数小于 5，检查点个数也小于 5，则判定为简单类型的测试用例，对于这类测试用例，可多选择一些用于实现自动化。

(2) 如果测试用例中包含的测试步骤在 5~15 个，检查点个数在 5~10 个，则判定为中等复杂类型的测试用例，对于这类测试用例，可选择少一些用于实现自动化。

(3) 如果测试用例中包含的测试步骤在 15~25 个，检查点个数在 10~15 个，则判定为复杂类型的测试用例，对于这类测试用例，可选择更少一些用于实现自动化。

至于选择的比例，则可参考一些项目的经验数值，或者根据项目实际情况自己调整。这种通过测试用例复杂度分组来筛选出自动化测试用例的方法比较简单易行，而又不失科学性。众所周知，自动化测试脚本的实现难度，在很大程度上取决于测试用例的复杂度，而测试用例的复杂度又在很大程度上取决于测试步骤和检查点的复杂度。

在自动化测试的测试用例范围内，找出每个测试用例的操作数量和验证点的数量，然后画出一个图表来找出平均值，并且定出控制点，这样可以基于被测试应用程序的实际情况而不是仅仅根据行业标准来判断复杂度。

例如，表 1.2 显示了 25 个测试用例中每个用例所包含的测试步骤。

表 1.2 25 个测试用例中每个用例所包含的测试步骤

测试用例序号	测试步骤
01	8
02	12
03	15
04	12
05	16
06	30
07	4

续表

测试用例序号	测试步骤
08	8
09	22
10	21
11	12
12	27
13	22
14	15
15	11
16	15
17	12
18	9
19	8
20	19
21	21
22	42
23	6
24	22
25	22

根据这个表，可以画出图 1.1 所示的复杂度与测试步骤个数的关系图。

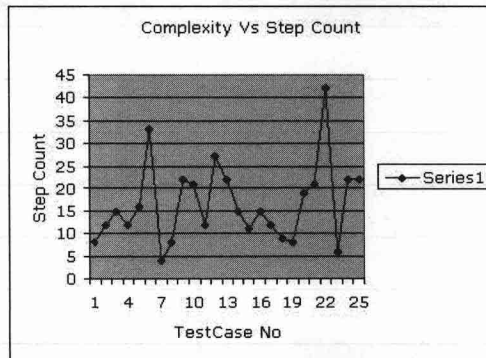


图 1.1 复杂度与测试步骤个数的关系图

然后，我们可以基于图 1.1 计算出平均的测试步骤个数是 16 个，以此为基准点，定出上、下限分别是 8 和 25，则可以这样定义测试用例的复杂度。

- 简单： ≤ 7 个步骤。
- 中等： ≥ 8 个步骤 $\sim \leq 16$ 个步骤。
- 复杂： ≥ 17 个步骤 $\sim \leq 25$ 个步骤。

类似的，我们可以再加入检查点的个数，按类似的方法进行计算。

1.3 对控件的熟悉程度与自动化测试成功实施之间的关系

我们这里讨论的是基于 GUI 的功能自动化测试。基于 GUI 的自动化测试工具都是基于 GUI 对象识别技术来设计的。基于 GUI 层面的测试需要与各种界面元素打交道，而且不同的编程语言和开发工具开发的应用程序在界面的表现、事件的响应上都略有不同。

对于自动化测试工程师而言，如果能充分了解不同控件的属性和方法的话，对于自动化测试的脚本开发会有很大的帮助。例如，对于 .NET 控件 TrackBar，普通的 QTP 测试工具的录制将得到如下脚本：

```
SwfWindow("Form1").SwfObject("trackBar1").Drag 13,12
SwfWindow("Form1").SwfObject("trackBar1").Drop 32,10
```

录制下来的是鼠标光标在 TrackBar 控件上的拖曳动作，这些脚本不仅可读性差，并且回放容易失败，不容易理解和维护。

如果熟悉 TrackBar 控件的属性和方法的话，则可以通过访问控件的内部属性来达到控制控件的目的。同样的操作在得到适当的处理之后将录制成如下脚本：

```
SwfWindow("Form1").SwfObject("trackBar1").SetValue 6
```

录制下来的是设置 TrackBar 控件位置值的脚本，这样脚本更容易理解，并且回放能准确定位到指定的 TrackBar 位置，增强了脚本的可维护性，这样的脚本要比前面的脚本更容易修改、更容易参数化。

1.4 自动化测试计划

在计划一个自动化测试项目时，必须考虑以下几个方面的内容。

- (1) 时间。

过早地用自动化测试会带来维护成本的增加。因为早期的程序界面一般不够稳定，处于频繁更改的状态，这时进行自动化测试往往得不偿失，疲于应付“动荡”的界面。

自动化测试不应该在界面尚未稳定的时候开始，但是，这并不意味着不需要计划和准备工作。在界面处于雏形时期，可以基于界面原型提供的控件来尝试自动化测试工具的适用性，因为有些控件是自动化测试工具不能识别和测试的。这时候，就要考虑工具的选择问题。

在开发人员着手开发一些核心的代码时，可能会同时开发出一些核心可重用的控件，而且是那种自定义的个性化控件，那么就需在这个阶段获得这些控件，并且尝试使用自动化工具来测试这些控件。如果发现有不适用的地方，则要考虑让开发人员重新设计控件，或者提供更多的测试接口。

(2) 人员。

自动化测试工程师应该具备一定的自动化测试基础知识，包括自动化测试工具的基础知识、自动化测试脚本的开发基础知识等；还需要了解各种测试脚本的编写和设计方法，知道在什么时候选取怎样的测试脚本开发方式，知道如何维护测试脚本；需要具备一定的编程技巧，熟悉某些测试脚本语言的基本语法和使用方法。

另外，自动化测试工程师与手工测试工程师一样，需要具备设计测试用例的基本方法和能力，具备软件涉及的基本业务的理解能力，以及具备将测试用例转换成自动化测试用例的能力。

(3) 工具。

大部分商业测试工具会指定某种语言，例如：WinRunner (TSL)、SilkTest (4test)、Robot (Test Basic)；但是，一些新的工具也开始使用标准语言，例如：QTP (VB Script)、XDE Tester (Java)。所以，在选择测试工具时要考虑这点。最好选择支持标准语言的测试工具，而且尽量与所在项目组的开发人员所使用并熟悉的语言一致。这样可以充分利用现有的编程知识和语言知识，而不需要花时间去熟悉厂商特定的语言（这些语言只能在这个工具上使用）。并且可以借助开发人员丰富的开发经验来协助进行测试脚本的设计和编写。

大部分商业测试工具没能很好地支持新的平台和很多第三方控件及个性化控件。例如，新的.NET 版本、操作系统，以及普遍使用的第三方控件，如 Component One、Infragistics、Janus 等。如果项目中使用这些较新的平台或大量使用这些第三方控件的话，就要小心选择测试工具了，否则会导致后面的脚本编写难度加大。建议在选用工具之前，充分评估并在项目的应用程序上试用。

例如，我们在一个测试项目就曾经对 TestComplete 和 QTP 两个工具进行了详细地比较，如表 1.3 所示。

表 1.3 TestComplete 与 QTP 的比较

功能特性	TestComplete	QTP
录制功能	支持	支持
操作系统	Microsoft Windows 98、Windows ME、Windows 2000、Windows XP (32 位和 64 位版本)、Windows Vista (32 位版本)、Windows Server 2003 (32 位和 64 位版本)或 Windows NT 4.0 与 Service Pack 6 或更新版本	Windows 2000—Service Pack 4 Update Rollup 1 for Windows 2000 Service Pack 4, Windows XP 32-Bit Edition—Service Pack 2 Windows XP 64-Bit Edition—Service Pack 2 Windows Server 2003 32-Bit Edition—Service Pack 2 Windows Server 2003 R2 (32-Bit x86), Windows 2003 64-Bit Edition, Windows Vista 32-bit Edition, Windows Vista 64-bit Edition—Service Pack 2
浏览器	Microsoft Internet Explorer ver. 5-7. Mozilla Firefox ver. 1.5.0.1-2.0. Any web browser based on the Microsoft WebBrowser control. Also, limited support is provided for Netscape Navigator ver. 8.1.2	Internet Explorer、Netscape、FireFox、AOL
数据库测试	BDE、ADO	DSN (ODBC32 Interface)
数据函数	DDT 对象支持 ADO、CSV、Excel	DataTable 对象支持 Excel
测试脚本	ProjectSuite 包含 Project, Project 包含脚本	测试脚本是一个文件夹, 包含一系列的支撑文件
代码视图	Code Explorer	Keyword View, Expert View
对象库	没有	对象库由单独文件管理和维护, 每个 Action 对应一个对象库, 对象库可导出并共享
Custom Classes	Object Mapping	Virutal Object Wizards
图像测试	支持	支持
错误恢复管理	Event	Recovery Manager
检查点	支持	支持
测试结果报告	专有的测试结果文件格式, 可导出为 XML 格式文件	保存为 XML 文件, 并可以转换成 HTML 格式文件
与测试管理工具的整合	可与 Microsoft Visual Studio 2005 Team System 或 AQdevTeam 整合	与 Quality Center 整合
调用 DLL	支持	支持

续表

功能特性	TestComplete	QTP
支持 Java	支持	支持
支持 Flex	不支持	Adobe 利用 QTP 的扩展性开发了 Flex 插件
支持 .NET	支持	支持
脚本语言	VBScript、DelphiScript、C++Script、C#Script、JScript	VBScript
支持脚本调试	支持	支持
易用性	中等	好
文档	充分	非常充分
最新版本	6.0	9.5
价格	便宜	贵
厂商	AutomatedQA	HP, 以前是 Mercury Interactive
曾用名	Aqtest	Astra QuickTest

总的来说,在为自动化测试项目做工具选型时,需要考虑以下几个方面的因素来决定选择哪个自动化测试工具:

- 对不同类型的应用程序和平台的支持;
- 对不同类型的操作系统的支持;
- 对不同的测试类型的支持;
- 脚本语言、编辑器和调试器;
- 录制测试脚本的能力;
- 应对变化的能力;
- 对控件和对象的支持;
- 支持不同渠道的测试数据;
- 运行测试与测试对象的同步;
- 检查点;
- 测试结果记录和导出报告;
- 扩展性;
- 测试多语言应用程序的能力;
- 对团队协作和源代码管理的支持;
- 对命令行和 OLE 自动化的支持;