中山大學
SUN YAT-SEN UNIVERSITY

Part II [Problem]
6. Test Automation

SE-307 Software Testing Techniques

http://my.ss.sysu.edu.cn/wiki/display/SE307/Home

Instructor: Dr. Wang Xinming, School of Software, Sun Yat-Sen University

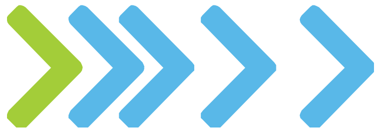
Review: Problems in Testing

- Fundamental problems
  - Test oracle （测试结果判定）
  - Test adequacy （测试充分性标准）
  - Test generation （测试数据生成）

- Important problems
  - Test automation （测试自动化）
  - Test management（测试计划和过程）

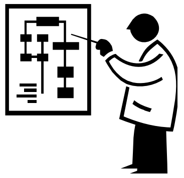
Test Automation（测试自动化）

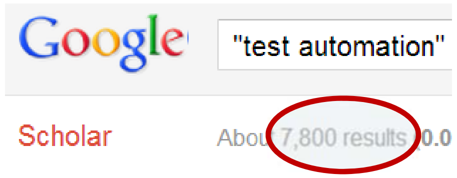
- Wikipedia: "*Test automation is the process of writing a computer program to do testing that would otherwise need to be done manually*"
  - *The test automation program control the setting up of test preconditions, the execution of test cases, and the comparison of actual outcomes to predicted outcomes.*
  - Provide efficient, repeatable, and consistent testing of the system under test。



Is Test Automation Important?

- From a **researcher** point of view:
  - No. Because it is not difficult.
    - Not a major problem as compared to the three fundamental problems.

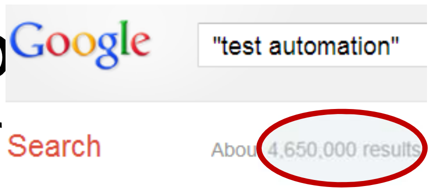




Google "test automation"

Scholar About 7,800 results 0.0

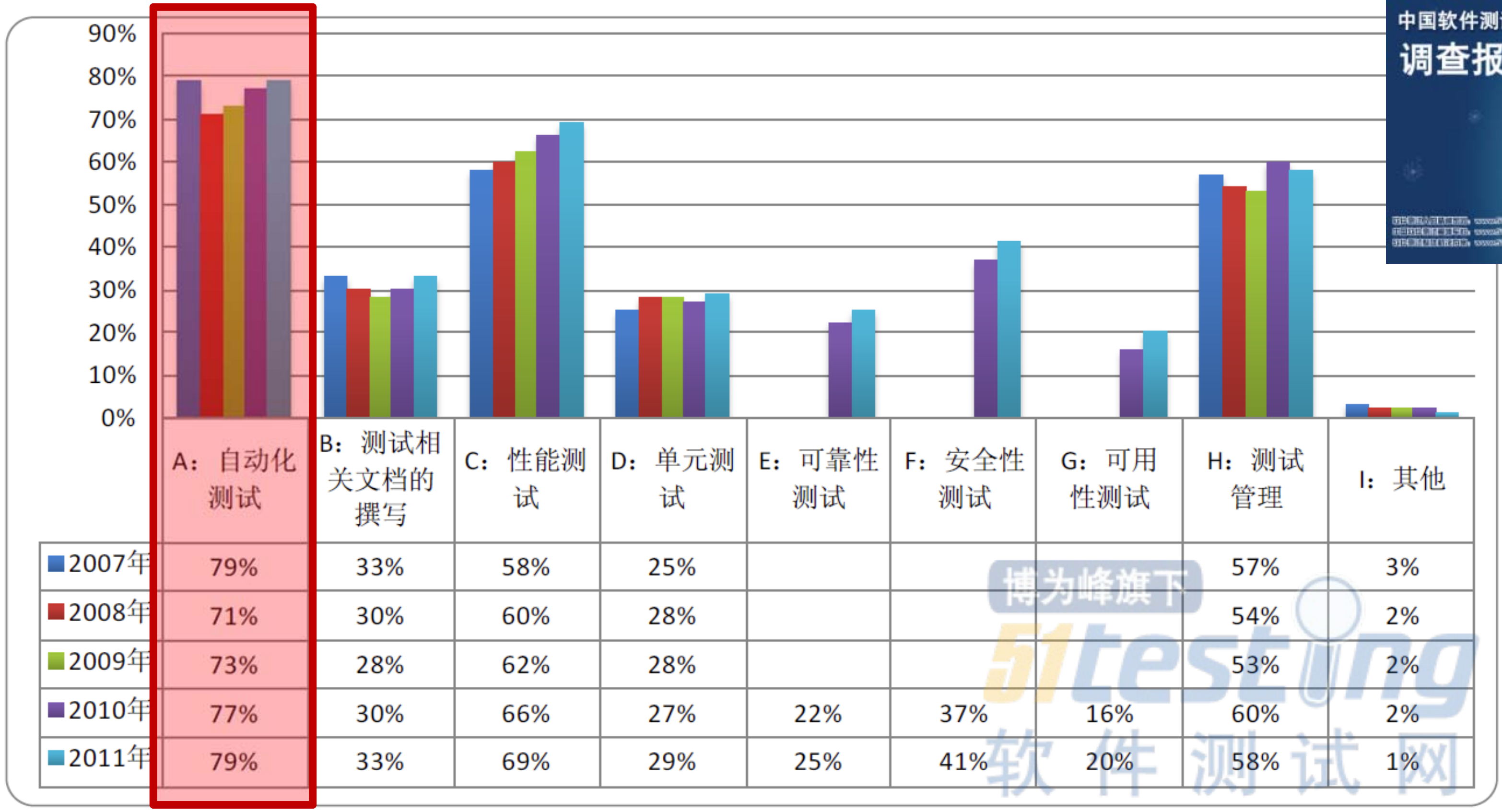
- From a **practitioner** point of view:
  - It is an very important problem:
    - Address several fundemental challenges in testing practice.



Google "test automation"

Search About 4,650,000 results

Is Test Automation Important?



	A：自动化测试	B：测试相关文档的撰写	C：性能测试	D：单元测试	E：可靠性测试	F：安全性测试	G：可用性测试	H：测试管理	I：其他
🟦2007年	79%	33%	58%	25%				57%	3%
🟥2008年	71%	30%	60%	28%				54%	2%
🟩2009年	73%	28%	62%	28%				53%	2%
🟪2010年	77%	30%	66%	27%	22%	37%	16%	60%	2%
🟦2011年	79%	33%	69%	29%	25%	41%	20%	58%	1%

历届调查中软件测试从业人员希望提高的软件测试技能

Understanding Test Automation

- Why?
  - Why do we want test automation?
- What?
  - What is test automation?
- When?
  - When is test automation recommended?
- How?
  - How to implement test automation?

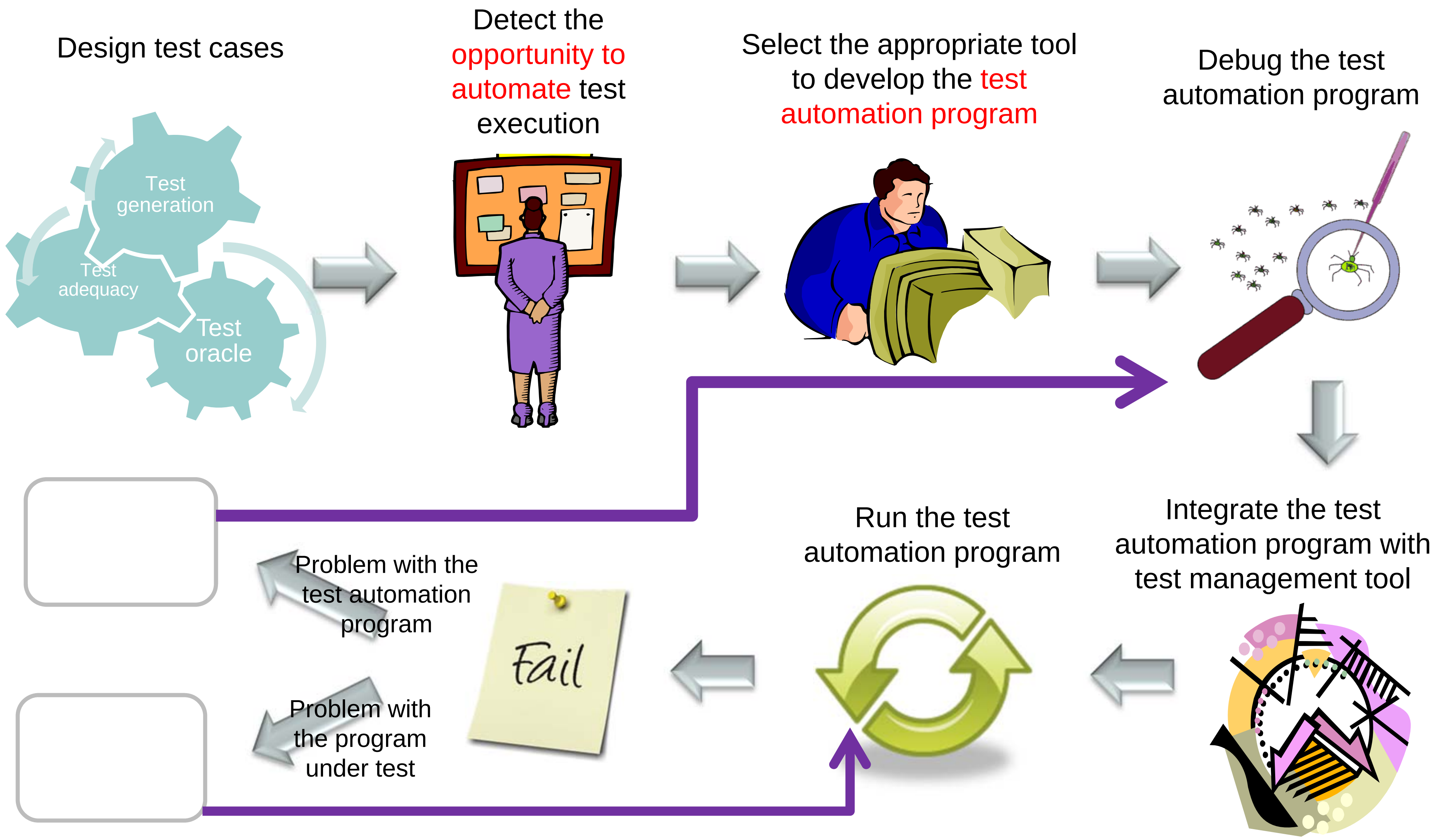
Why Test Automation?

- Testing needs to cover multiple platforms and releases in a short time.
  - Very tedious to repeat the test execution.
- Test execution demands a huge amount of time and attention.
  - Involve millions of mouse clicks and key presses.
  - And testers need to get everything right.
- Test result record and reporting is error-prone.
  - Inconsistency with the actual result.

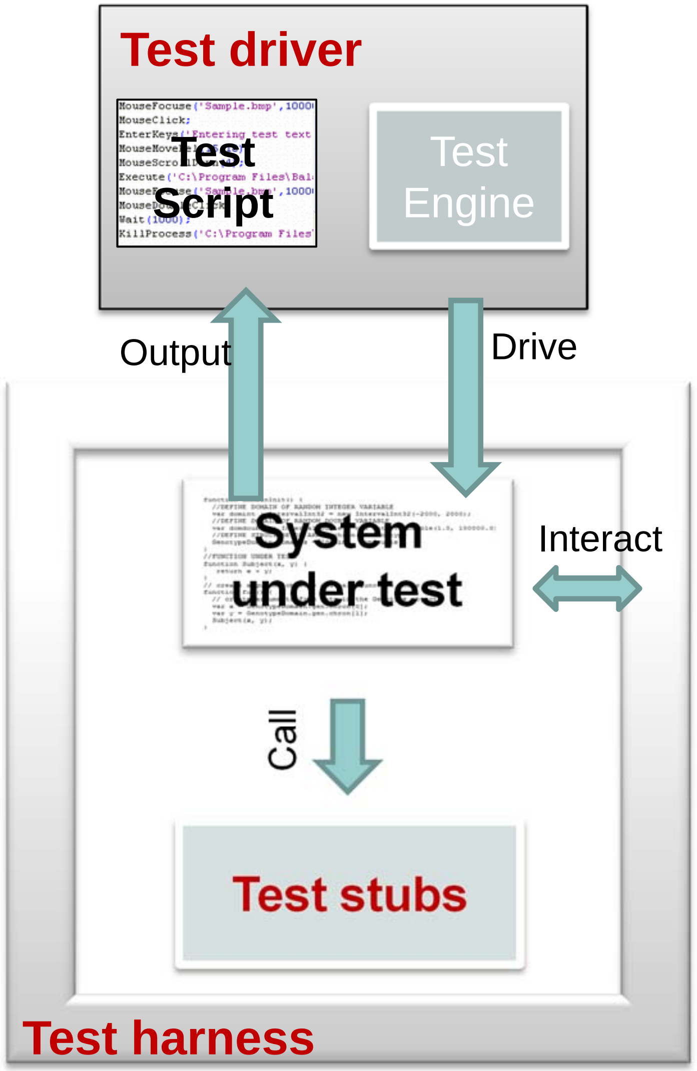
Understanding Test Automation

- Why?
  - Why we need test automation?
- What?
  - What is test automation?
- When?
  - When is test automation recommended?
- How?
  - What are the techniques to implement test automation?

Test Automation Process


Design test cases
Test generation
Test adequacy
Test oracle
Detect the opportunity to automate test execution
Select the appropriate tool to develop the test automation program
Debug the test automation program
Integrate the test automation program with test management tool
Run the test automation program
Fail
Problem with the test automation program
Problem with the program under test

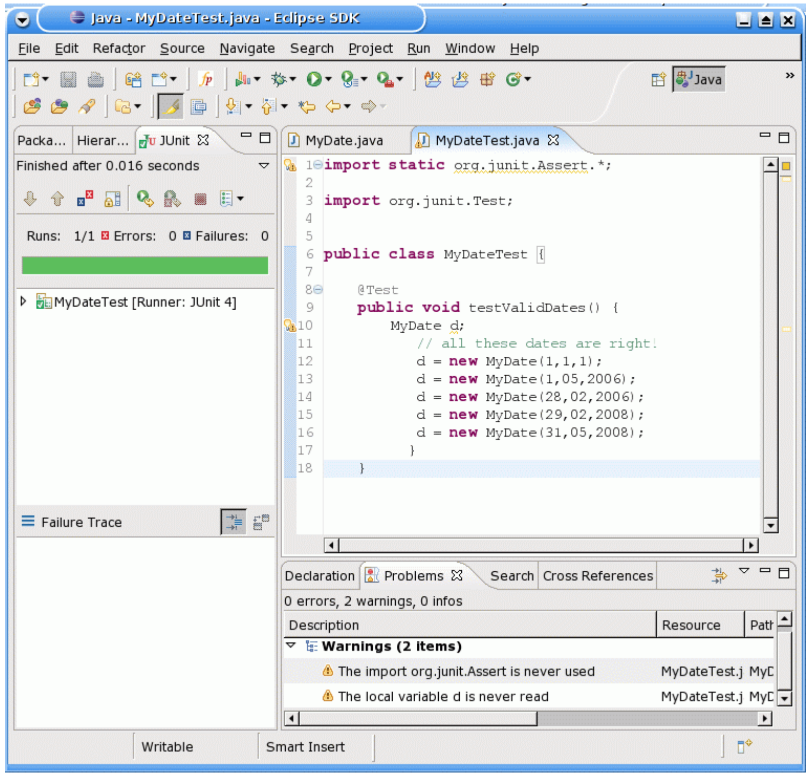
Test Automation Program



- Include three parts:
  - Test driver （测试驱动）
    - Drive test execution and implement test oracle
    - Consist of test engine and test scripts when using test automation tool (e.g. RFT).
  - Test stub （测试桩）
    - Substitute for functions/methods/objects/components/library that the code under test depends on .
  - Test harness （测试套）
    - Substitutes for other parts of the deployed environment (e.g. software simulation of a hardware device)

Automation Opportunity

Objectives	Purpose	Automation opportunity	Automation Techniques
Functional testing, unit level (单元测试)	Verify that the unit is correctly implemented.	**Always**	Unit Test Framework (e.g. JUnit) and IDE (e.g. Eclipse)


Java - MyDateTest.java - Eclipse SDK
File Edit Refactor Source Navigate Search Project Run Window Help
Java
Packa... Hierar... JUnit
MyDate.java MyDateTest.java
Failure Trace
Declaration Problems Search Cross References
Writable Smart Insert

Finished after 0.016 seconds

Runs: 1/1 Errors: 0 Failures: 0

MyDateTest [Runner: JUnit 4]

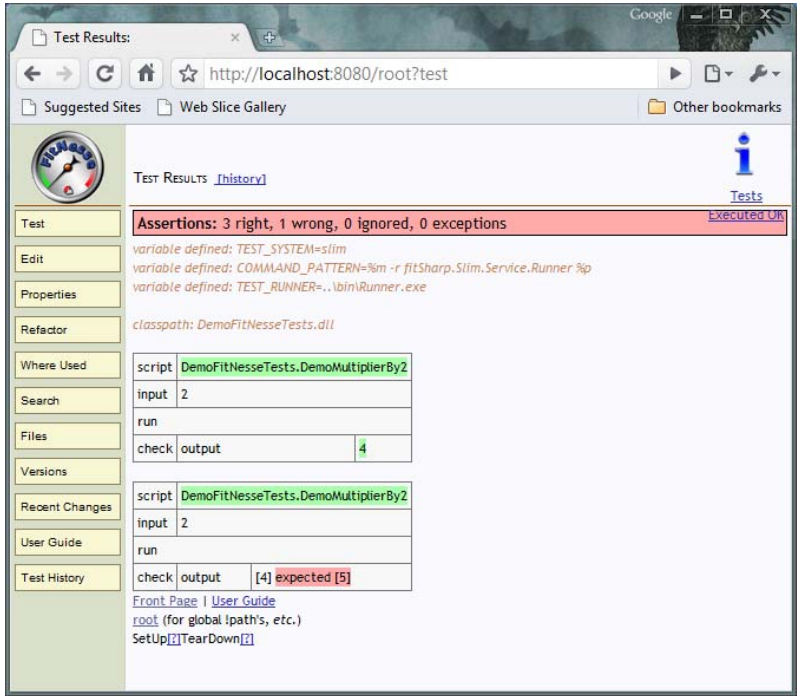
```java
import static org.junit.Assert.*;

import org.junit.Test;


public class MyDateTest {

    @Test
    public void testValidDates() {
        MyDate d;
            // all these dates are right!
            d = new MyDate(1,1,1);
            d = new MyDate(1,05,2006);
            d = new MyDate(28,02,2006);
            d = new MyDate(29,02,2008);
            d = new MyDate(31,05,2008);
        }
    }
```

0 errors, 2 warnings, 0 infos

Description	Resource	Path
Warnings (2 items)		
The import org.junit.Assert is never used	MyDateTest.j	MyD
The local variable d is never read	MyDateTest.j	MyD

Automation Opportunity

Objectives	Purpose	Automation opportunity	Automation Techniques
Functional testing, unit level (单元测试)	Verify that the unit is correctly implemented.	**Always**	Unit Test Framework (e.g. JUnit) and IDE (e.g. Eclipse)
Functional testing, integration level (集成测试)	Verify that the units interact correctly.	**Always**	API testing, FitNesse



Test Results [history]

Assertions: 3 right, 1 wrong, 0 ignored, 0 exceptions

variable defined: TEST_SYSTEM=slim
variable defined: COMMAND_PATTERN=%m -r fitSharp.Slim.Service.Runner %p
variable defined: TEST_RUNNER=..\bin\Runner.exe

classpath: DemoFitNesseTests.dll

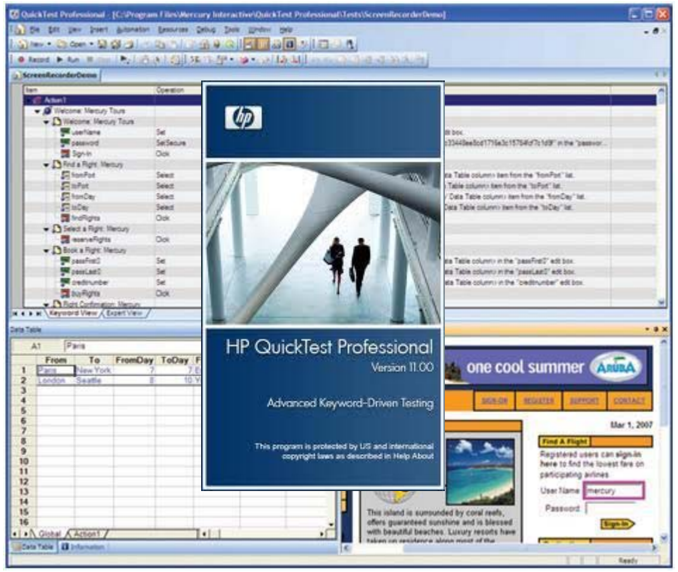
script	DemoFitNesseTests.DemoMultiplierBy2	
input	2	
run		
check	output	4

script	DemoFitNesseTests.DemoMultiplierBy2	
input	2	
run		
check	output	[4] expected [5]

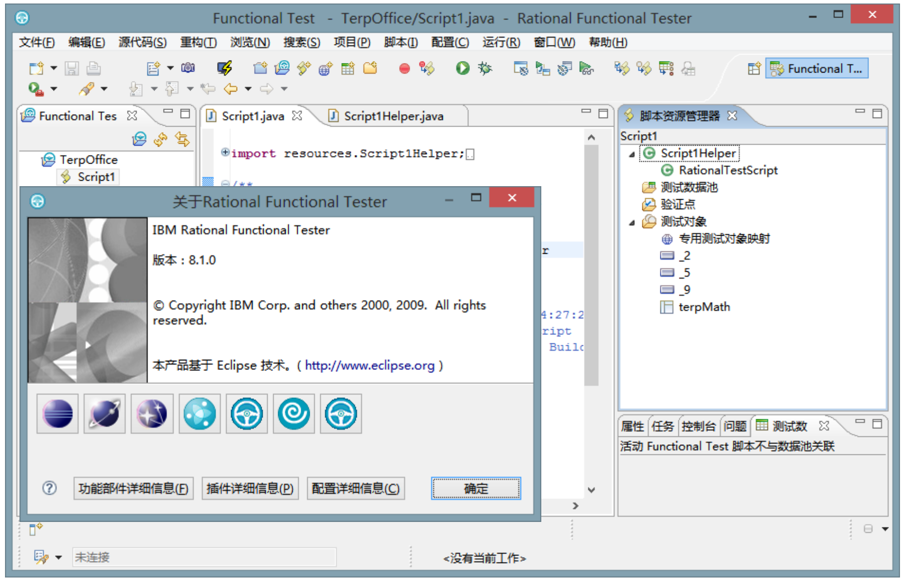
Front Page | User Guide
root (for global !path's, *etc.*)
SetUp[?]TearDown[?]

Automation Opportunity

Objectives	Purpose	Automation opportunity	Automation Techniques
Functional testing, unit level (单元测试)	Verify that the unit is correctly implemented.	**Always**	Unit Test Framework (e.g. JUnit) and IDE (e.g. Eclipse)
Functional testing, integration level (集成测试)	Verify that the units interact correctly.	**Always**	API testing, FitNesse
Functional testing, system level (系统测试)	Verify that the system correctly implement the functionality specified by the requirement	**High**	Capture/replay, Scripting, Data-driven, Keyword-driven



HP QuickTest Professional
Version 11.00
Advanced Keyword-Driven Testing
This program is protected by US and international copyright laws as described in Help About
one cool summer



```
import resources.Script1Helper;
```

IBM Rational Functional Tester

版本：8.1.0

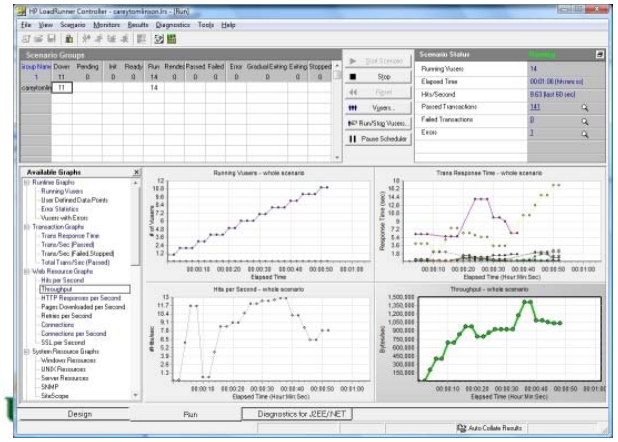
© Copyright IBM Corp. and others 2000, 2009. All rights reserved.

本产品基于 Eclipse 技术。(http://www.eclipse.org)

活动 Functional Test 脚本不与数据池关联

Automation Opportunity

Objectives	Purpose	Automation opportunity	Automation Techniques
Functional testing, unit level (单元测试)	Verify that the unit is correctly implemented.	**Always**	Unit Test Framework (e.g. JUnit) and IDE (e.g. Eclipse)
Functional testing, integration level (集成测试)	Verify that the units interact correctly.	**Always**	API testing, FitNesse
Functional testing, system level (系统测试)	Verify that the system correctly implement the functionality specified by the requirement	**High**	Capture/replay, Scripting, Data-driven, Keyword-driven
Load testing （并发测试）	Verify the behavior of the system under both normal and anticipated peak load conditions.	**Always**	Capture/replay, Scripting Thread Pool, Distributed run
Stress testing （压力测试）	Verify the behavior of the system beyond its capacity.	**Always**	Capture/replay, Scripting Thread Pool, Distributed run


HP LoadRunner Controller
Scenario Groups
Scenario Status
Available Graphs
Running Vusers - whole scenario
Trans Response Time - whole scenario
Hits per Second - whole scenario
Throughput - whole scenario
Design
Run
Diagnostics for J2EE/.NET

Automation Opportunity

Objectives	Purpose	Automation opportunity	Automation Techniques
Functional testing, unit level (单元测试)	Verify that the unit is correctly implemented.	**Always**	Unit Test Framework (e.g. JUnit) and IDE (e.g. Eclipse)
Functional testing, integration level (集成测试)	Verify that the units interact correctly.	**Always**	API testing, FitNesse
Functional testing, system level (系统测试)	Verify that the system correctly implement the functionality specified by the requirement	**High**	Capture/replay, Scripting, Data-driven, Keyword-driven
Load testing （并发测试）	Verify the behavior of the system under both normal and anticipated peak load conditions.	**Always**	Capture/replay, Scripting Thread Pool, Distributed run
Stress testing （压力测试）	Verify the behavior of the system beyond its capacity.	**Always**	Capture/replay, Scripting Thread Pool, Distributed run
Reliability testing （可靠性测试）	Verify that the software does not fail frequently under normal usage scenarios.	**Always**	Capture/replay, Execution profile

Automation Opportunity

Objectives	Purpose	Automation opportunity	Automation Techniques
Functional testing, unit level (单元测试)	Verify that the unit is correctly implemented.	**Always**	Unit Test Framework (e.g. JUnit) and IDE (e.g. Eclipse)
Functional testing, integration level (集成测试)	Verify that the units interact correctly.	**Always**	API testing, FitNesse
Functional testing, system level (系统测试)	Verify that the system correctly implement the functionality specified by the requirement	**High**	Capture/replay, Scripting, Data-driven, Keyword-driven
Load testing （并发测试）	Verify the behavior of the system under both normal and anticipated peak load conditions.	**Always**	Capture/replay, Scripting Thread Pool, Distributed run
Stress testing （压力测试）	Verify the behavior of the system beyond its capacity.	**Always**	Capture/replay, Scripting Thread Pool, Distributed run
Reliability testing （可靠性测试）	Verify that the software does not fail frequently under normal usage scenarios.	**Always**	Capture/replay, Execution profile
Usability testing （可用性测试）	Verify that the ease of using and learning the software.	**None**	

Automation Opportunity

Objectives	Purpose	Automation opportunity	Automation Techniques
Functional testing, unit level (单元测试)	Verify that the unit is correctly implemented.	**Always**	Unit Test Framework (e.g. JUnit) and IDE (e.g. Eclipse)
Functional testing, integration level (集成测试)	Verify that the units interact correctly.	**Always**	API testing, FitNesse
Functional testing, system level (系统测试)	Verify that the system correctly implement the functionality specified by the requirement	**High**	Capture/replay, Scripting, Data-driven, Keyword-driven
Load testing （并发测试）	Verify the behavior of the system under both normal and anticipated peak load conditions.	**Always**	Capture/replay, Scripting Thread Pool, Distributed run
Stress testing （压力测试）	Verify the behavior of the system beyond its capacity.	**Always**	Capture/replay, Scripting Thread Pool, Distributed run
Reliability testing （可靠性测试）	Verify that the software does not fail frequently under normal usage scenarios.	**Always**	Capture/replay, Execution profile
Usability testing （可用性测试）	Verify that the ease of using and learning the software.	**None**	
Acceptance testing （确认测试）	Verify that the software delivers what the users expect (alpha testing and beta testing).	**Low**	

Understanding Test Automation

- Why?
  - Why we need test automation?
- What?
  - What is test automation?
- When?
  - When is test automation recommended?
- How?
  - What are the techniques to implement test automation?

Test Automation: "IS" and "IS NOT"

- Test automation **is** software development
  - Test automation is a product: software to test software.
  - Require analysis, design, implementation, and maintenance, etc.
  - Need money, time, people, and skills.
- Test automation **is** long time investment
  - Upfront costs: tool support, learning, development.
  - Maintenance costs: update obsolete test scripts.
  - Need to consider Return on Investment

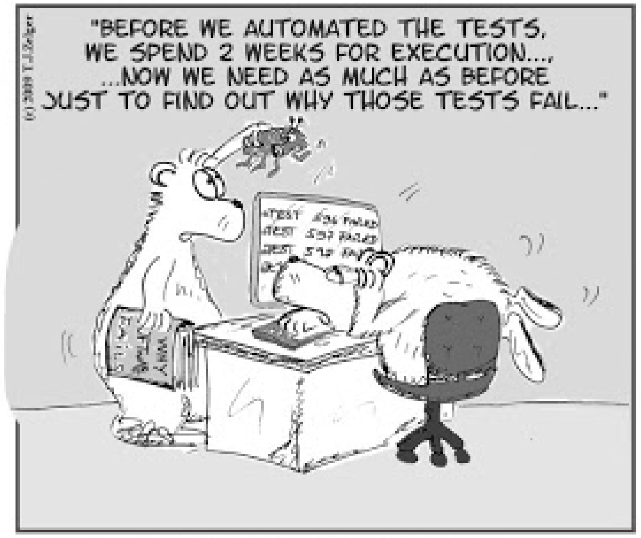
I CAN'T DEVELOP AN AUTOMATED TESTING SYSTEM BY THE ARBITRARY DEADLINE YOU SET.
TRY WORKING SMARTER, NOT HARDER, WITH A SENSE OF URGENCY, AND A BIAS FOR ACTION.
OR MAYBE YOU COULD DO SOMETHING DIFFERENTLY.
I'M NOT THE ONE WHO CAN'T GET HIS WORK DONE.
YOU ELIMINATED THE BUDGET FOR AUTOMATED TEST SOFTWARE. HOW ARE WE GOING TO TEST OUR NEW CODE?
GO WRITE SOME AUTOMATED TEST SOFTWARE, YOU BIG BABY. I ALREADY PAY YOU, SO IT'S FREE.
www.dilbert.com
scottadams@aol.com
© 2007 Scott Adams, Inc./Dist. by UFS, Inc.

Repeat: Test Automation is Development

- Windows NT 4.0 had 6 million lines of code, and 12 million lines of test code
- Poor design and programming practices for automated testing:
  - Embedded constants
  - No modularity
  - No source control
  - No documentation
  - No requirements analysis

Test Automation: “IS” and “IS NOT”

- Test automation is not silver bullet for testing
  - Still need to address the three fundamental problems.
  - Test automation might result in poor flexibility.
  - Test automation only exposes limited types of bugs.
- Test automation is not always recommended
  - Low execution frequency of test cases.
  - Fragile design.
  - Involve operation on physical devices.
  - Validation testing.
- Test automation is not complete replacement of manual testing
  - Not appropriate for validation testing.
  - Not cost-effective in some scenarios.


"BEFORE WE AUTOMATED THE TESTS, WE SPEND 2 WEEKS FOR EXECUTION..., ...NOW WE NEED AS MUCH AS BEFORE JUST TO FIND OUT WHY THOSE TESTS FAIL..."

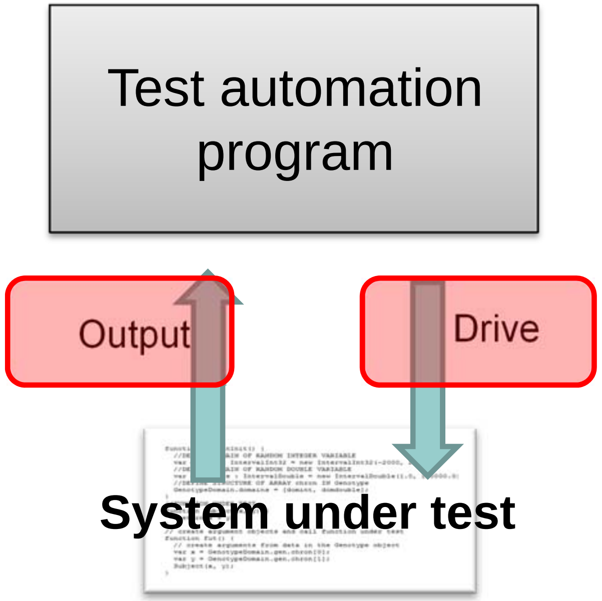
Test Automation side effect

When is Automation Recommended?

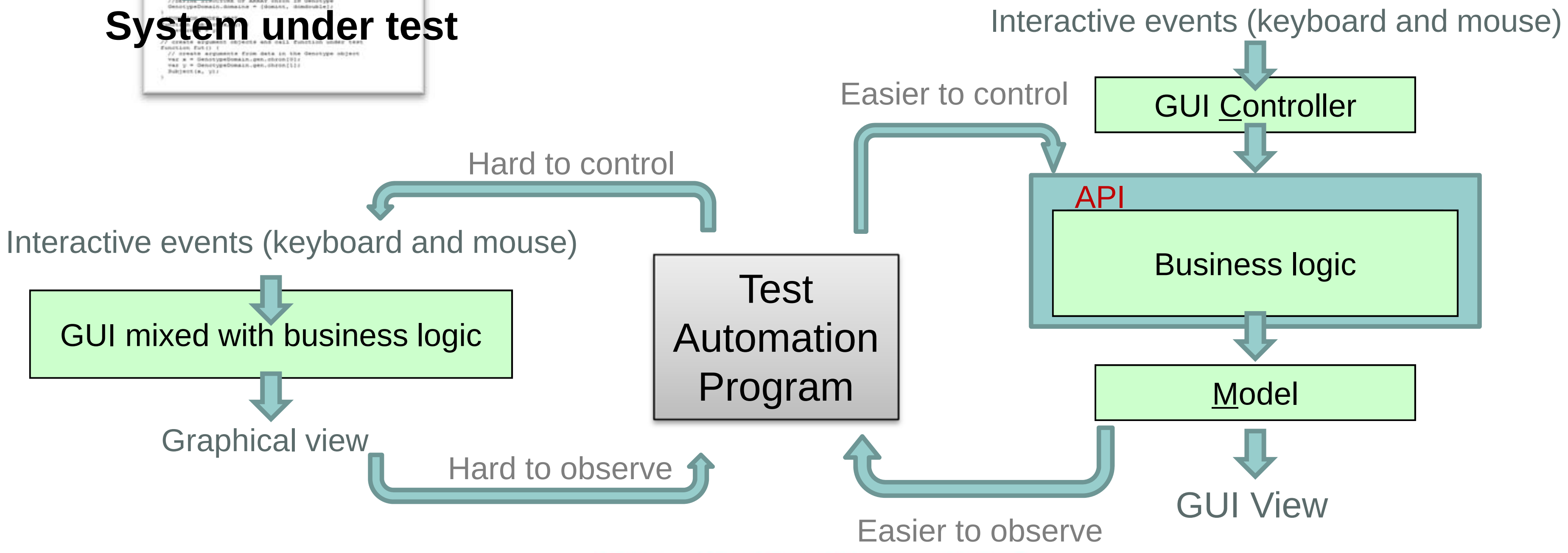
- The benefits of test automation need to be greater than the costs of automation.
- General rule of thumb: it is expected that tests will have to be run many times
  - **Regression** functional testing
  - Load / stress testing
  - Reliability testing

  - “Agile” development process
  - Continuous integration
  - Test-Driven development

Factors that Affect Cost (Testability)



- **Controllability (可控制性)**
  - Can the test automation program easily control the program execution?
- **Observability (可观察性)**
  - Can the test automation program easily observe the output of the program?



Design for Testability

- Approaches to improve controllability and observability
  - Apply separation of concerns in the design
    - Separate application logic from GUI
  - Simulation of external dependencies with test stub and test harness.
    - e.g. Hardware device, network, database
  - Implement configuration options to trigger corner cases.
    - e.g. Out of heap memory, hard disk out of space.
  - Use assertions
    - Expose internal state errors
  - Make program failures obvious
    - Keep the output simple and well formatted.
    - Log failures to a separate file

Understanding Test Automation

- Why?
  - Why we need test automation?
- What?
  - What is test automation?
- When?
  - When is test automation recommended?
- How?
  - How to implement test automation?

General Practice

- Design test cases (recall our lectures on test adequacy).

- Start with a known state.
  - Decide on the execution environment (e.g. OS, database)
  - Record this environment

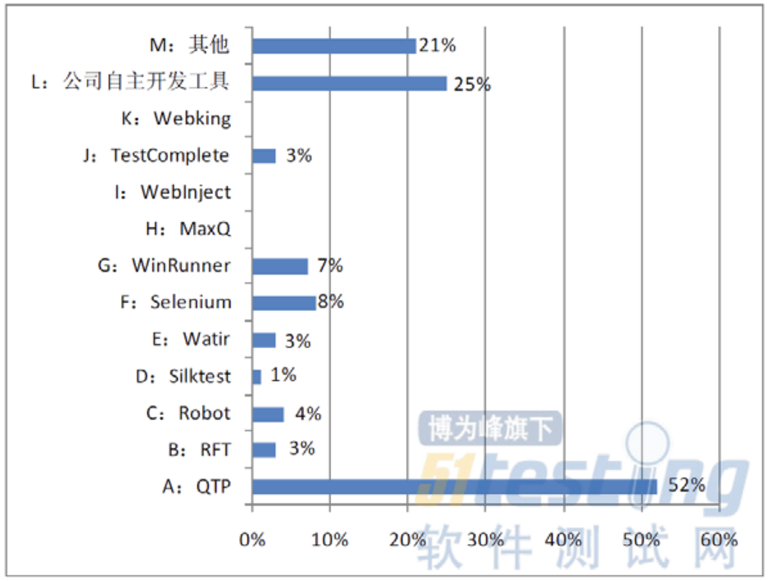
- Develop test scripts.
  - Then run them and observe their output.

- Check for errors (recall our lectures on test oracle).
  - Then implement the analysis into the test scripts.

Test Automation Techniques

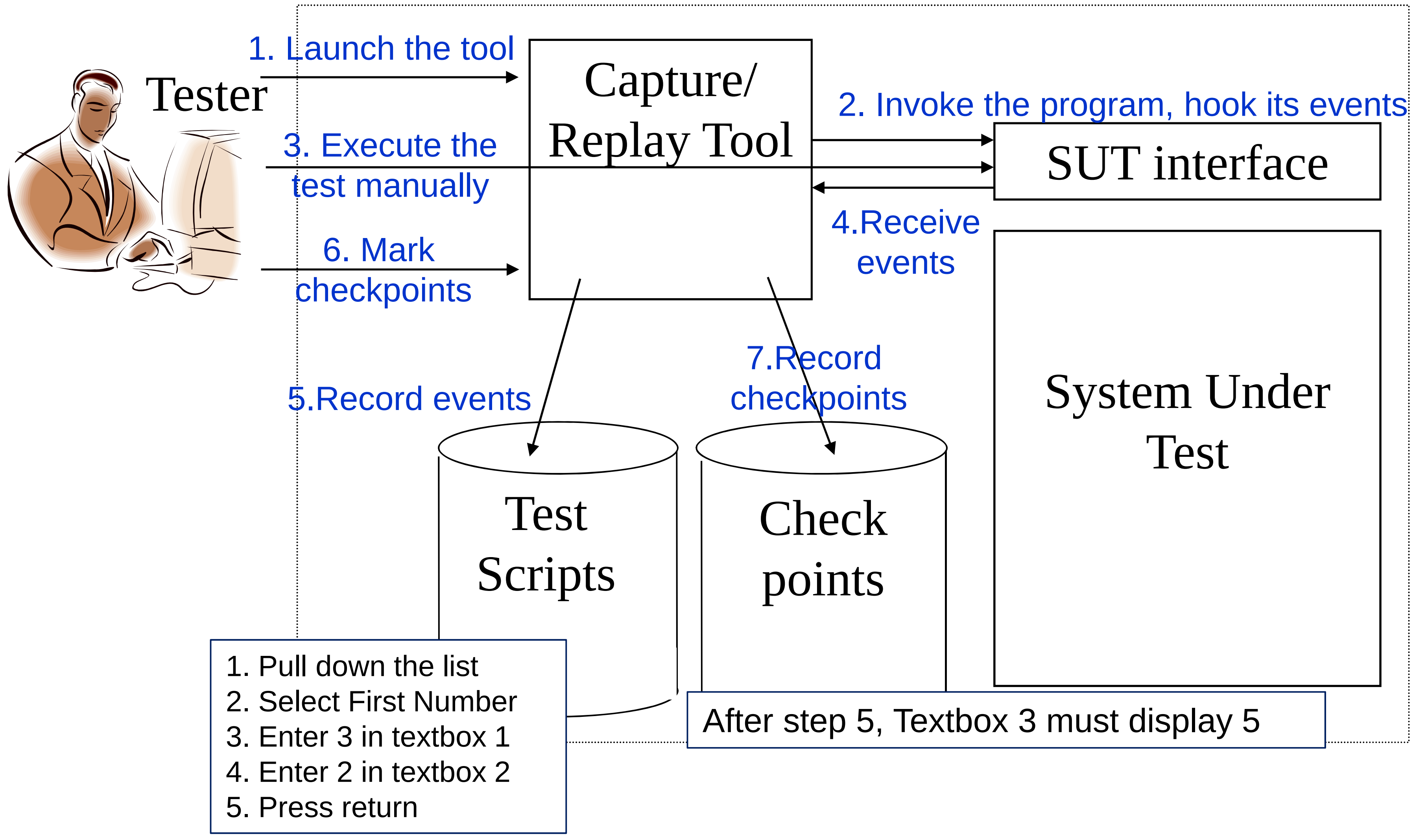
- Capture/replay
- Structural Test Scripts Development
- Data-driven test automation
- Keyword-driven test automation

- Mainstream Tools:
  - HP QuickTest Professional (QTP)
  - IBM Rational Functional Test (RFT)
  - AutomatedQA TestComplete
  - Borland SilkTest
  - Marathon



2011 年调查中软件测试从业人员常用的功能自动化测试工具分布

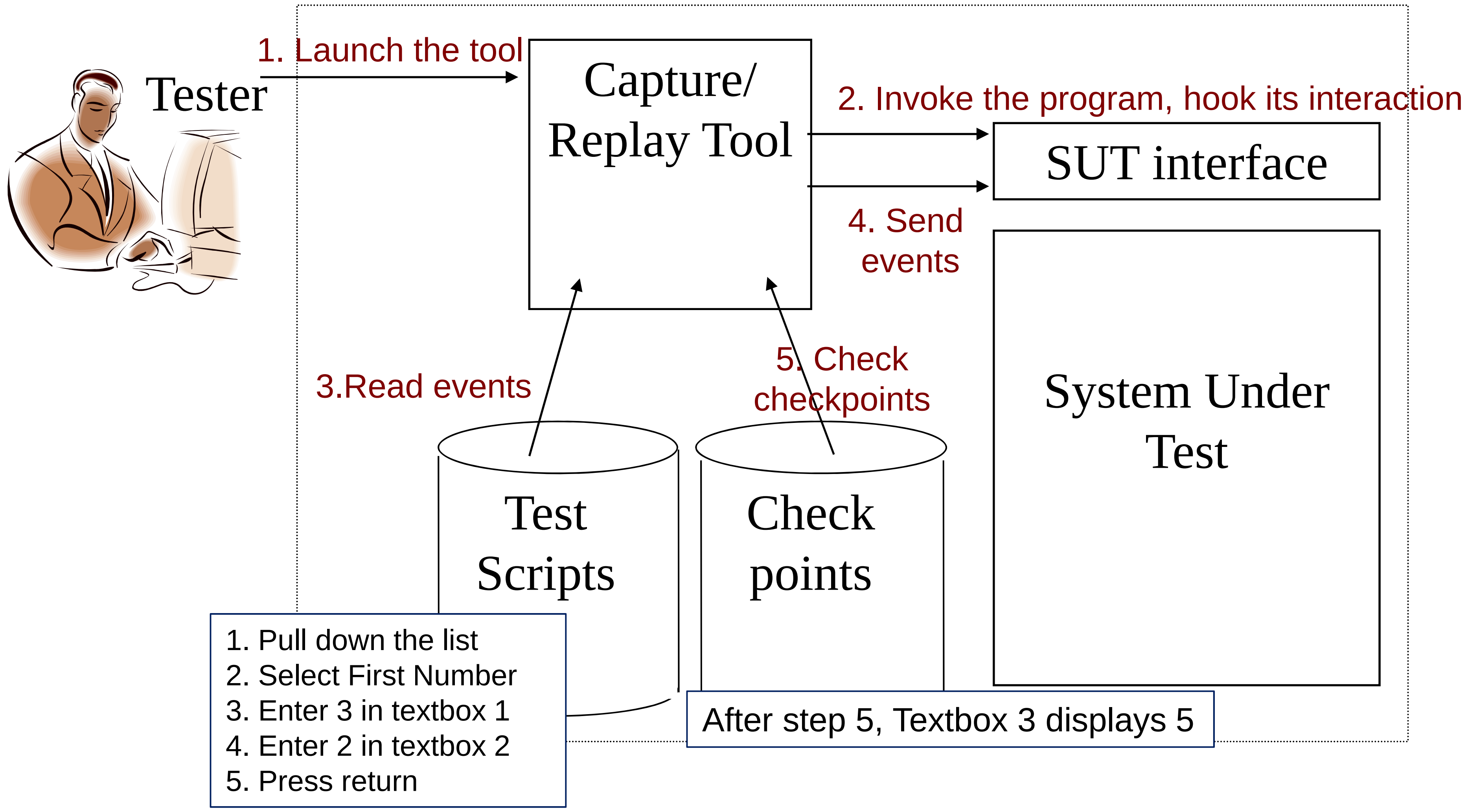
Capture Test Scripts



1. Pull down the list
2. Select First Number
3. Enter 3 in textbox 1
4. Enter 2 in textbox 2
5. Press return

After step 5, Textbox 3 must display 5

Replay Test Scripts



1. Pull down the list
2. Select First Number
3. Enter 3 in textbox 1
4. Enter 2 in textbox 2
5. Press return

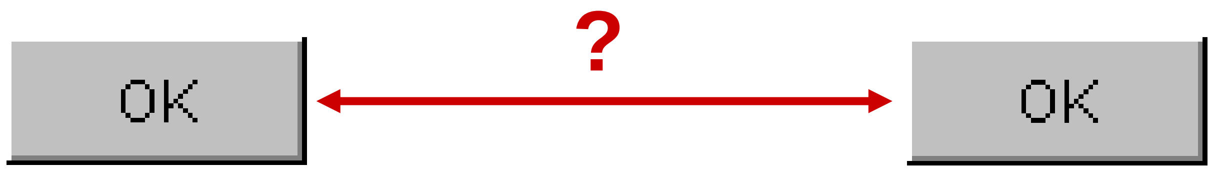
After step 5, Textbox 3 displays 5

Test Scripts

- Simulate the operations on the system under test by the users or another system.
  - Command line inputs
  - GUI events
  - Network events
  - …
- Can use any imperative languages
  - **RFT**: Java
  - **QTP**: VBScript
  - **TestComplete**: VBScript or JScript
  - **SilkTest**: C++
  - **Marathon**: Python or Jython

Key Issues in Test Scripts

- Identification of GUI objects
  - If the tool finds a button during script replay, how does it know that this is the same button it found when recording this script?



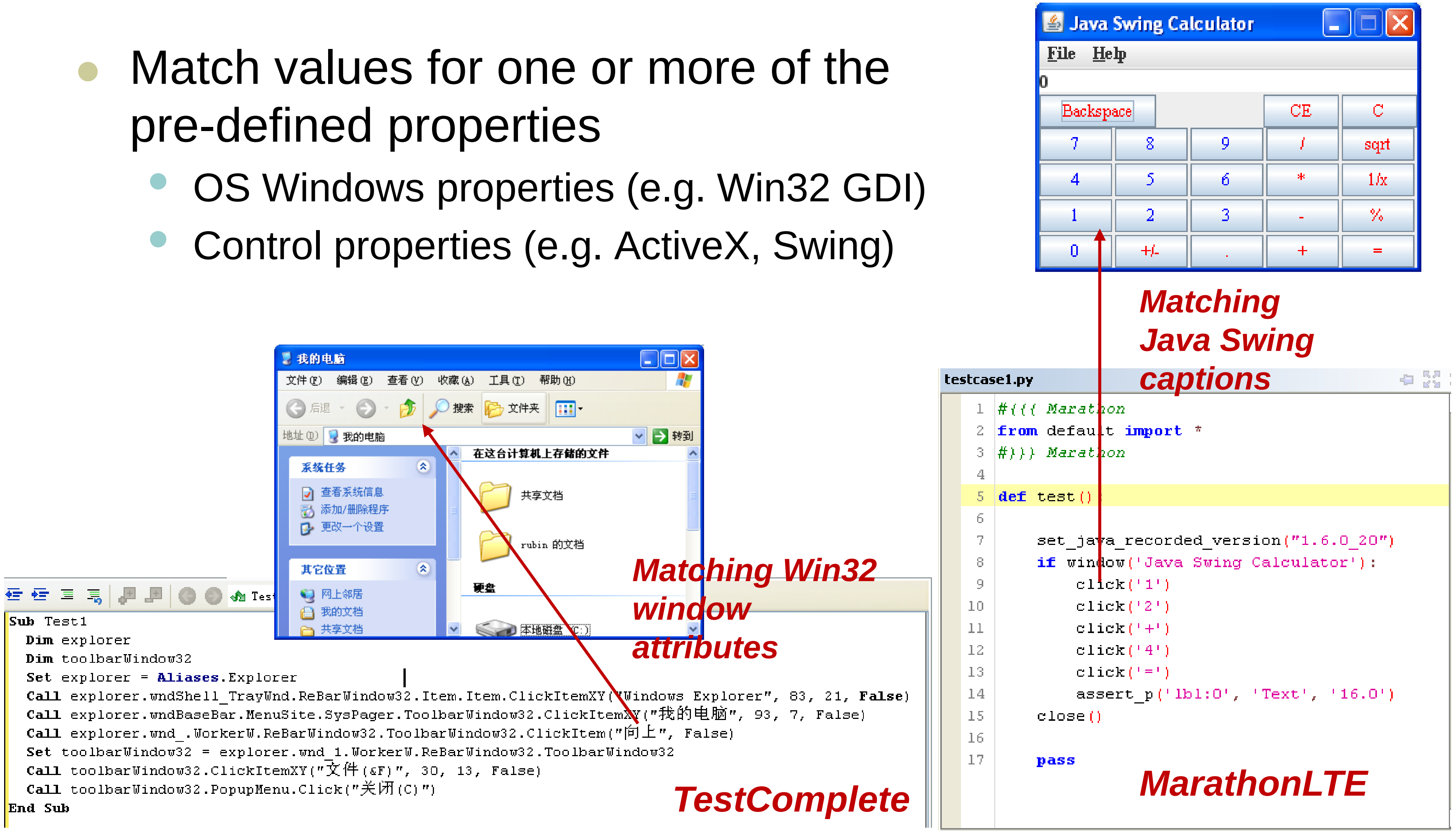
- Test oracle
  - How to check expected output?

Issue 1: GUI objects Identification

- Simple way: ***find-by-attribute***
  - Finding objects by matching the values of their properties (e.g. caption, background color, etc.)
  - Put the attribute names and values directly into the test script.
  - Many open source test automation tools only support this way, such as MarathonLTE.
- More general way: ***object map***
  - An intermediate layer that maps each GUI object identifier to a *descriptor*.
    - Find-by-attributes is a special case.
    - Kept in a separated file.
  - Most advanced test automations tool support this way: QTP (called *object repository*), RFT (called *test object map*), TestComplete (called *name mapping*), Marathon Commercial version (called *object map*)

Find-by-Attributes

- Match values for one or more of the pre-defined properties
  - OS Windows properties (e.g. Win32 GDI)
  - Control properties (e.g. ActiveX, Swing)

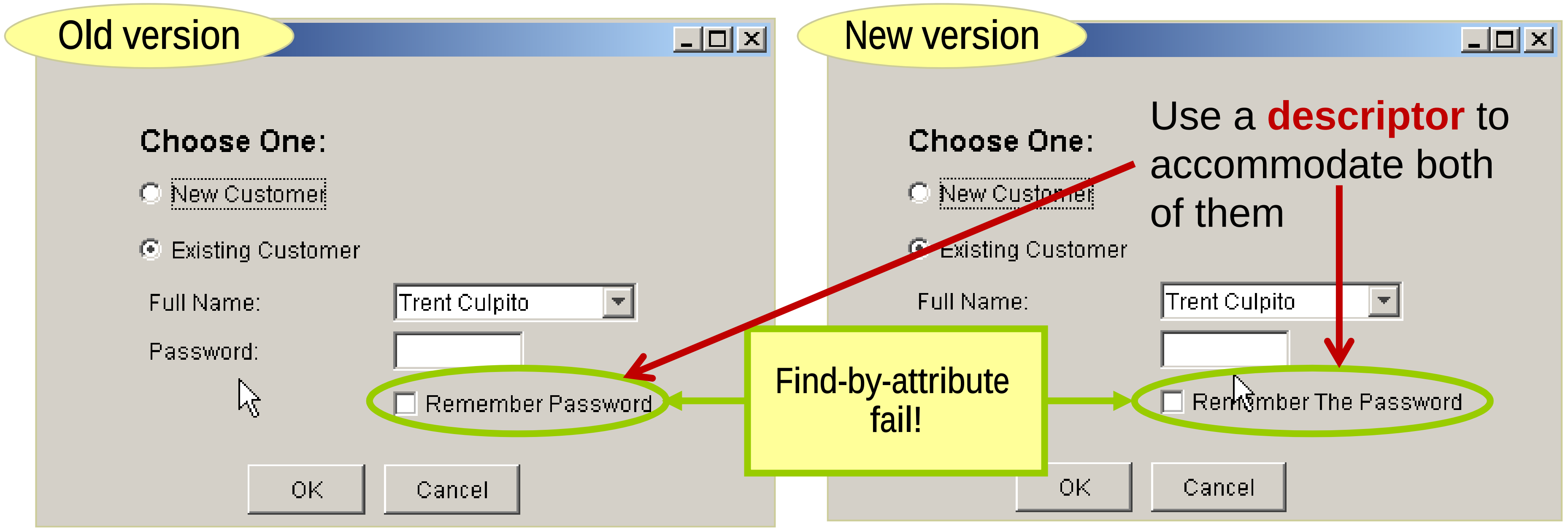


```
Sub Test1
  Dim explorer
  Dim toolbarWindow32
  Set explorer = Aliases.Explorer
  Call explorer.wndShell_TrayWnd.ReBarWindow32.Item.Item.ClickItemXY("Windows Explorer", 83, 21, False)
  Call explorer.wndBaseBar.MenuSite.SysPager.ToolbarWindow32.ClickItemXY("我的电脑", 93, 7, False)
  Call explorer.wnd_.WorkerW.ReBarWindow32.ToolbarWindow32.ClickItem("向上", False)
  Set toolbarWindow32 = explorer.wnd_1.WorkerW.ReBarWindow32.ToolbarWindow32
  Call toolbarWindow32.ClickItemXY("文件(&F)", 30, 13, False)
  Call toolbarWindow32.PopupMenu.Click("关闭(C)")
End Sub
```

```
#{{{ Marathon
from default import *
#}}} Marathon

def test():

    set_java_recorded_version("1.6.0_20")
    if window('Java Swing Calculator'):
        click('1')
        click('2')
        click('+')
        click('4')
        click('=')
        assert_p('lbl:0', 'Text', '16.0')
    close()

    pass
```

Object Map: Why Do We Need it?

- Properties of the objects can be frequently changed during updates.
- Even if the program has not been changed, the properties of an object can still change in different test runs
  - e.g. Captions changes with locales: “open” vs. “打开”



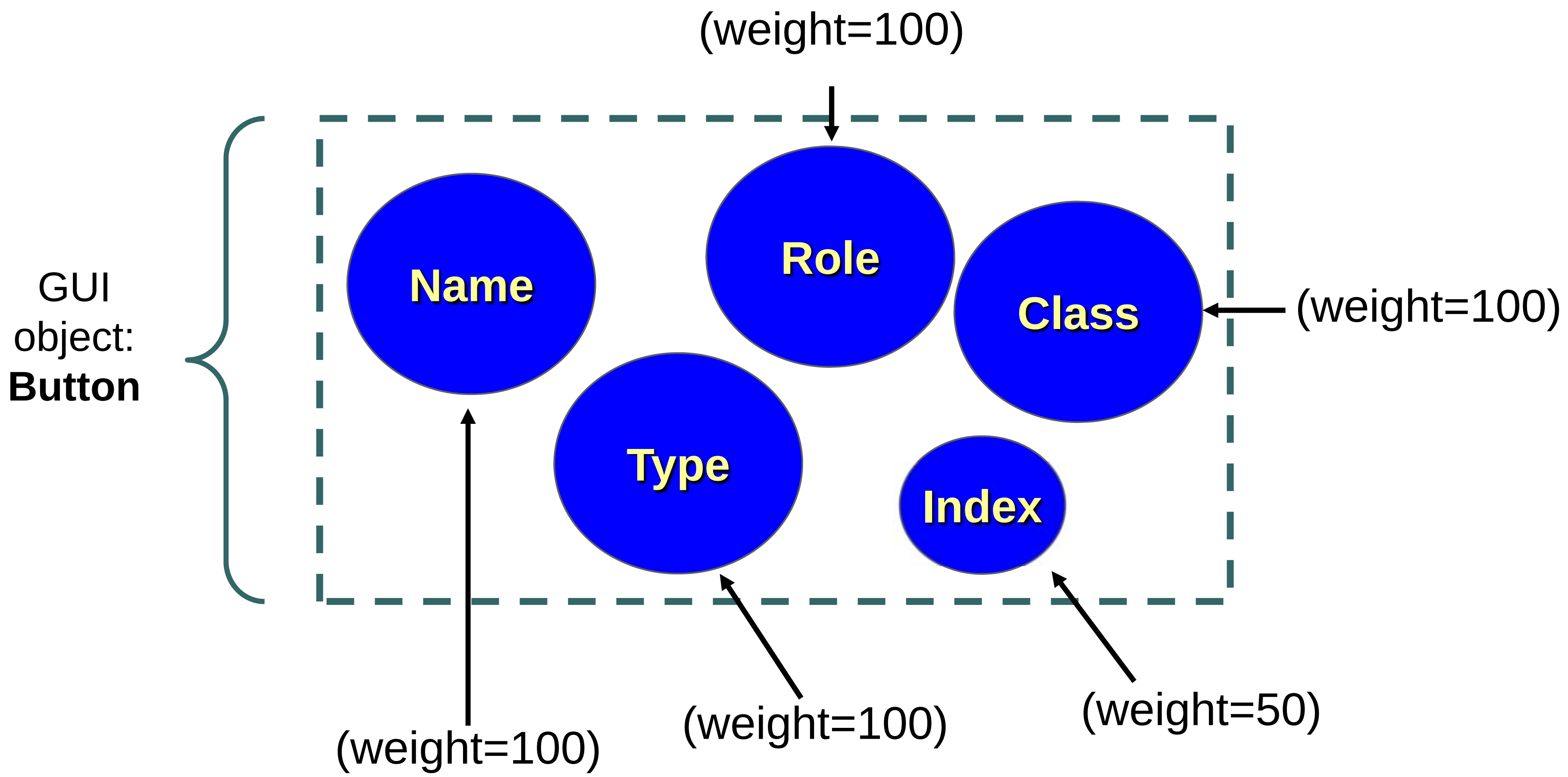
Descriptor Types

- Weighted multiple properties
- Indexing
- Patterns
- Virtual objects

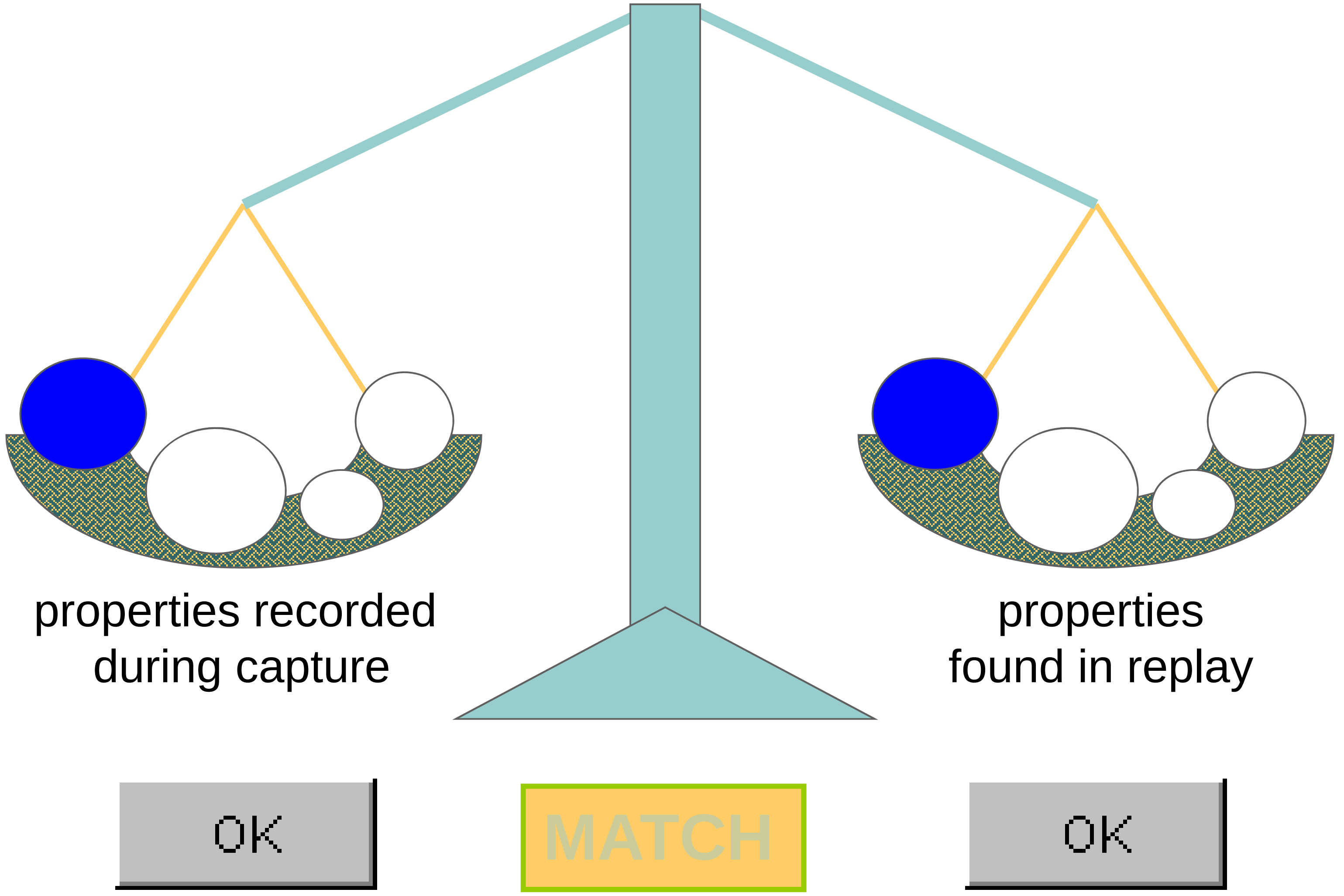
Weighted multiple properties

- How do you recognize your girlfriend/boyfriend in a crowd?
  - Most important property: Face
  - Second most important property: Voice
  - Third most important property: Cloth
  - …
- The same idea applied to identify GUI objects
  - Put different “weights” to the properties.

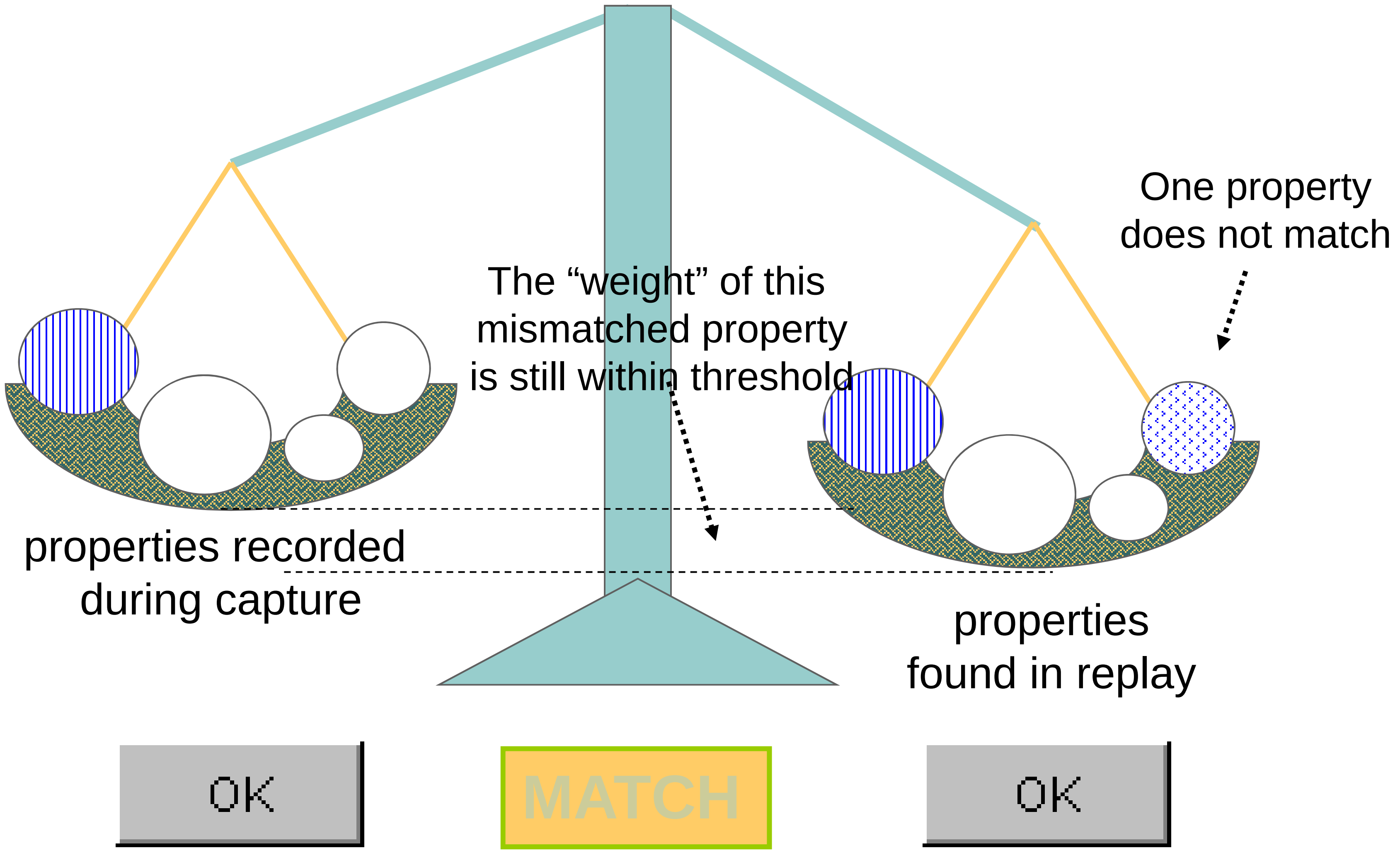
Example


(weight=100)
GUI object: **Button**
Name
Role
Class
(weight=100)
Type
Index
(weight=100)
(weight=100)
(weight=50)

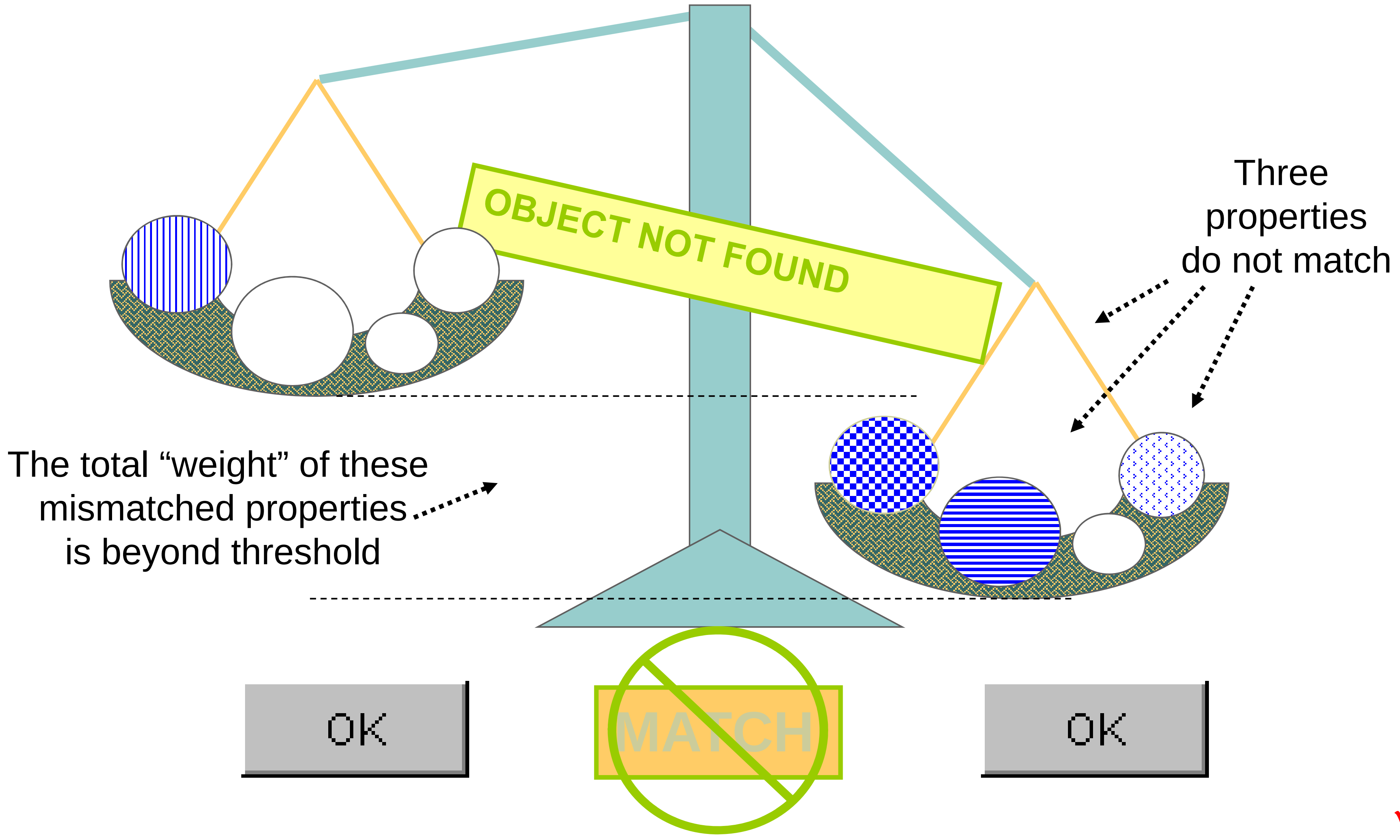
Exact Match


properties recorded during capture
properties found in replay
OK
MATCH
OK

Close-Enough Match


One property does not match
The “weight” of this mismatched property is still within threshold
properties recorded during capture
properties found in replay
OK
MATCH
OK

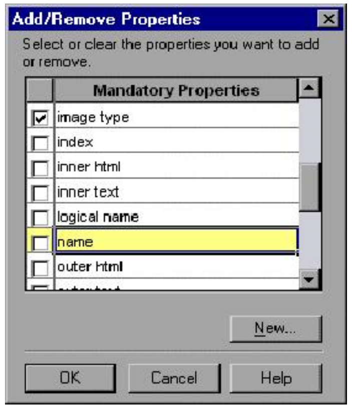
Too Many Differences: Object Not Found

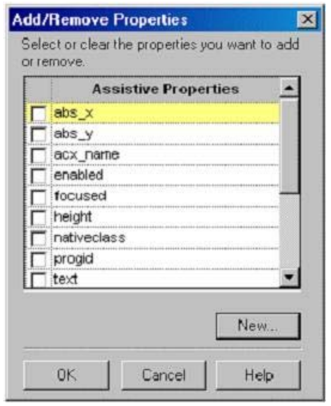

OBJECT NOT FOUND
Three properties do not match
The total "weight" of these mismatched properties is beyond threshold
OK
MATCH
OK

✓

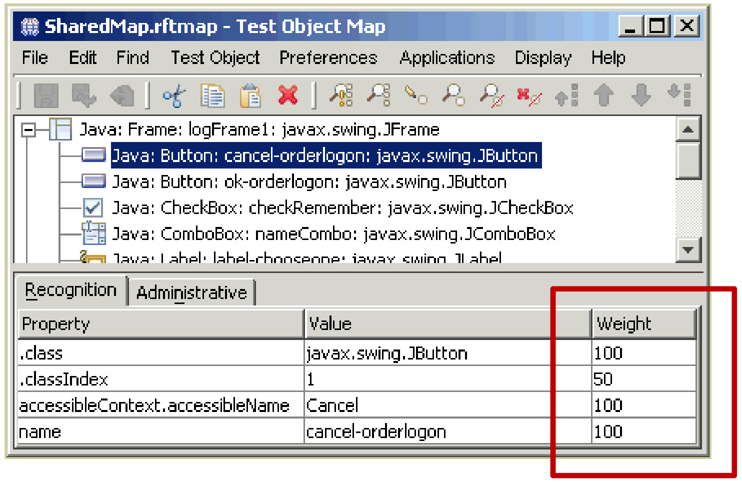
Tools Implementation

- **QTP** allows two levels of weights for properties: mandatory and assistive





- **RFT** allows weights to be set at 1~100


SharedMap.rftmap - Test Object Map
File Edit Find Test Object Preferences Applications Display Help
Java: Frame: logFrame1: javax.swing.JFrame
Java: Button: cancel-orderlogon: javax.swing.JButton
Java: Button: ok-orderlogon: javax.swing.JButton
Java: CheckBox: checkRemember: javax.swing.JCheckBox
Java: ComboBox: nameCombo: javax.swing.JComboBox
Recognition
Administrative

Property	Value	Weight
.class	javax.swing.JButton	100
.classIndex	1	50
accessibleContext.accessibleName	Cancel	100
name	cancel-orderlogon	100

Descriptor Types

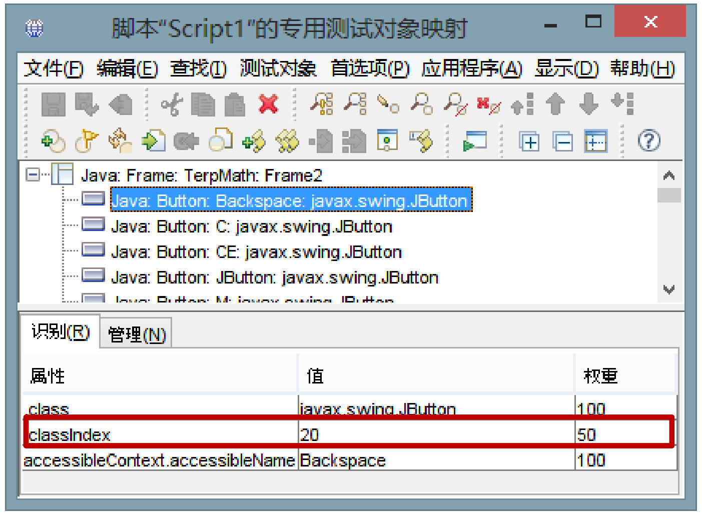
- Weighted multiple properties
- Indexing
- Patterns
- Virtual objects

Indexing

- Give each GUI object an index based on its relation with the other GUI objects in the same window.
  - **Creation-order:** the order in which the GUI object is created in the window relative to other objects in the same window.
    - e.g. the first button created in the window has an index 0, the second one has an index 1, etc.
  - **Location-order**: Indicates the order in which the GUI object is shown within the window relative to other objects.
    - e.g. the top-left button in the window has an index 0, the one below it has an index 1, etc.

Tools Implementation

- **QTP** supports both ways (called **ordinal identifier**)
  - **Creation-order**: QTP will assign a value to INDEX property of an object. The value is based on the order in which the object appears within the source code.
  - **Location-order**: QTP will assign a value to LOCATION property of an object. The value is based on the order in which the object appears within the window. Values are assigned in columns from top to bottom, and left to right.
- **RFT** supports creation-order only.
  - Use the property classIndex
  - The order in which the GUI object is created.


脚本"Script1"的专用测试对象映射
文件(F) 编辑(E) 查找(I) 测试对象 首选项(P) 应用程序(A) 显示(D) 帮助(H)
识别(R)
管理(N)

Java: Frame: TerpMath: Frame2
Java: Button: Backspace: javax.swing.JButton
Java: Button: C: javax.swing.JButton
Java: Button: CE: javax.swing.JButton
Java: Button: JButton: javax.swing.JButton

属性	值	权重
class	javax.swing.JButton	100
classIndex	20	50
accessibleContext.accessibleName	Backspace	100

Descriptor Types

- Weighted multiple properties
- Indexing
- Patterns
- Virtual objects

Use Patterns to Match Attributes

- Regular expression
  - Matches the values of string properties
- Numeric range
  - Specifies a range of numeric values to match number properties

Regular Expression

Acceptable values:	**Regular expression:**
customer Customer	**[cC]**ustomer
red blue green	red**|**blue**|**green
Remember Password Remember the Password	Remember**[the]*** Password (or) Remember **.*** Password

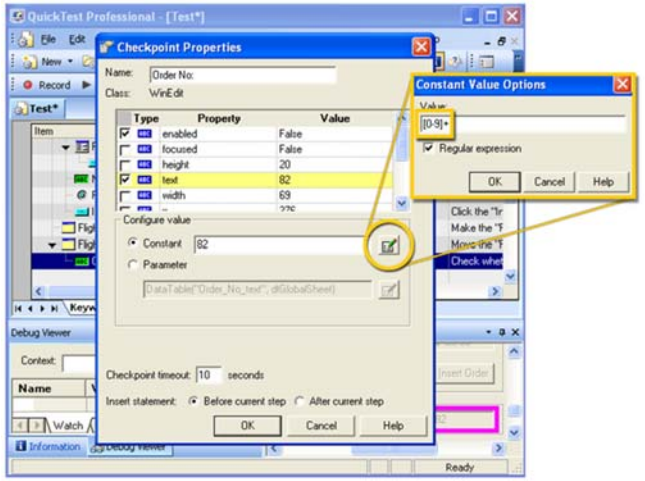
Numeric Range

Specify the lower and upper bounds of the numeric range

Acceptable values:	**Numeric range:**
Any number between 1 and 15, including 1 and 15	**[**1 **..** 15**]**
Any number between 1 and 15, including 1 but **not including** 15	**[**1 **..** 15**>**

Tools Implementation

- QTP: supports regular expression



QuickTest Professional - [Test*]

Checkpoint Properties

Name: Order No:
Class: WinEdit

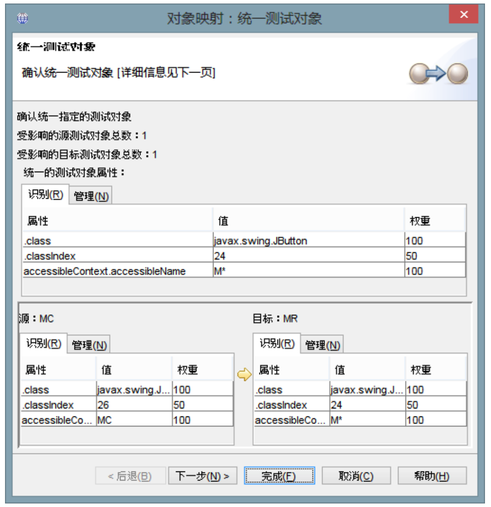
	Type	Property	Value
✓		enabled	False
		focused	False
		height	20
✓		text	82
		width	69

Configure value
(● Constant) 82
Parameter DataTable("Order_No_text", dtGlobalSheet)

Checkpoint timeout: 10 seconds
Insert statement: (● Before current step) After current step

Constant Value Options
Value: [0-9]*
✓ Regular expression

- RFT: supports both regular expression and numeric range



对象映射：统一测试对象

统一测试对象

确认统一测试对象 [详细信息见下一页]

确认统一指定的测试对象
受影响的源测试对象总数：1
受影响的目标测试对象总数：1
统一的测试对象属性：

属性	值	权重
.class	javax.swing.JButton	100
.classIndex	24	50
accessibleContext.accessibleName	M*	100

源：MC

属性	值	权重
.class	javax.swing.J...	100
.classIndex	26	50
accessibleCo...	MC	100

目标：MR

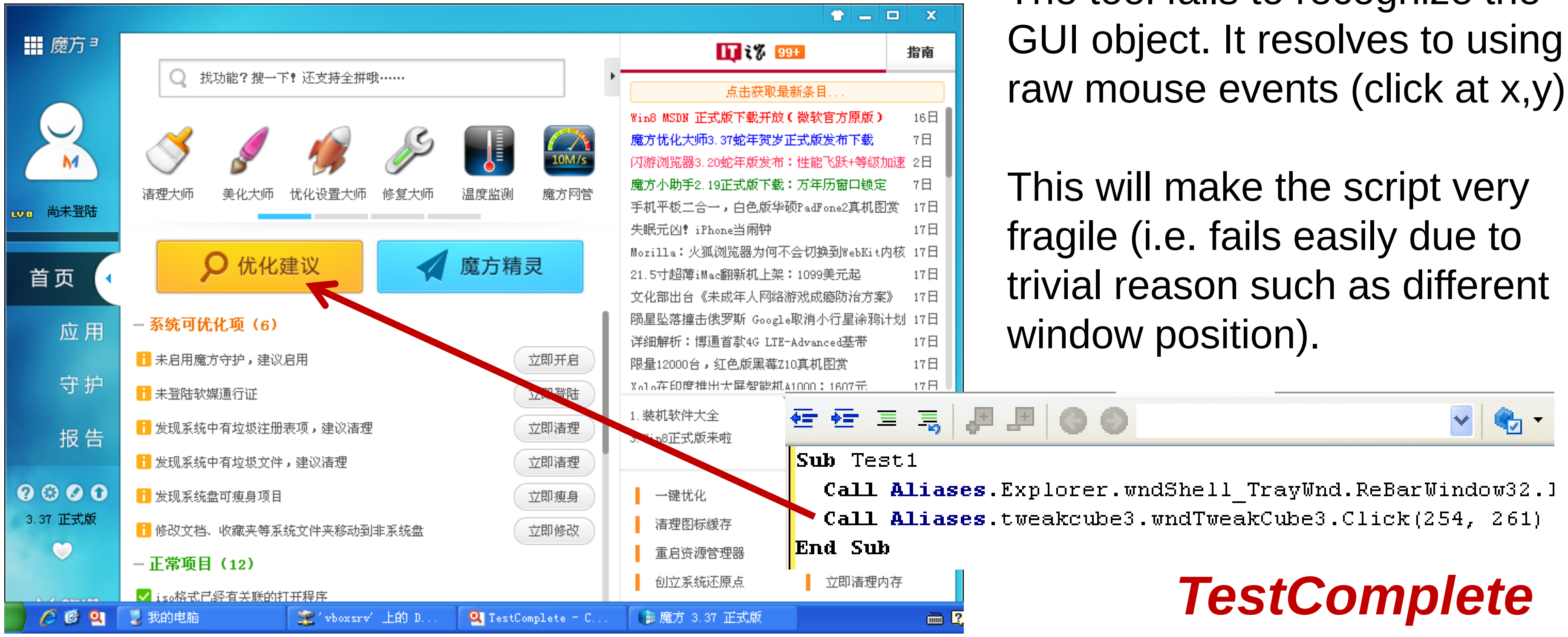
属性	值	权重
.class	javax.swing.J...	100
.classIndex	24	50
accessibleCo...	M*	100

Descriptor Types

- Weighted multiple attributes
- Indexing
- Patterns
- Virtual objects

Virtual Objects

- **Situation**: your programs use customized GUI objects not recognized by the automation tool.
- **Example**:



— 系统可优化项（6）
- 未启用魔方守护，建议启用
- 未登陆软媒通行证
- 发现系统中有垃圾注册表项，建议清理
- 发现系统中有垃圾文件，建议清理
- 发现系统盘可瘦身项目
- 修改文档、收藏夹等系统文件夹移动到非系统盘

— 正常项目（12）
- iso格式已经有关联的打开程序

Win8 MSDN 正式版下载开放（微软官方原版）	16日
魔方优化大师3.37蛇年贺岁正式版发布下载	7日
闪游浏览器3.20蛇年版发布：性能飞跃+等级加速	2日
魔方小助手2.19正式版下载：万年历窗口锁定	7日
手机平板二合一，白色版华硕PadFone2真机图赏	17日
失眠元凶！iPhone当闹钟	17日
Mozilla：火狐浏览器为何不会切换到WebKit内核	17日
21.5寸超薄iMac翻新机上架：1099美元起	17日
文化部出台《未成年人网络游戏成瘾防治方案》	17日
陨星坠落撞击俄罗斯 Google取消小行星涂鸦计划	17日
详细解析：博通首款4G LTE-Advanced基带	17日
限量12000台，红色版黑莓Z10真机图赏	17日
Xolo在印度推出大屏智能机A1000：1607元	17日

1.装机软件大全
3.Win8正式版来啦

```
Sub Test1
  Call Aliases.Explorer.wndShell_TrayWnd.ReBarWindow32.]
  Call Aliases.tweakcube3.wndTweakCube3.Click(254, 261)
End Sub
```

The tool fails to recognize the GUI object. It resolves to using raw mouse events (click at x,y)

This will make the script very fragile (i.e. fails easily due to trivial reason such as different window position).

TestComplete

Virtual Objects

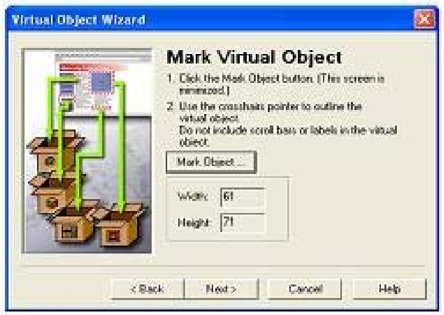
- In normal recording if we select the paint brush window, the script will look like following:

```
Window(“魔方”).Activate
Window(“魔方”).Click(72,16)
Window(“魔方”).Close
```

- If we are using virtual object, the script will look like following:

```
Window(“魔方”).Activate
Window(“魔方”).Virtual Button(“RED”)
Window(“魔方”).Close
```

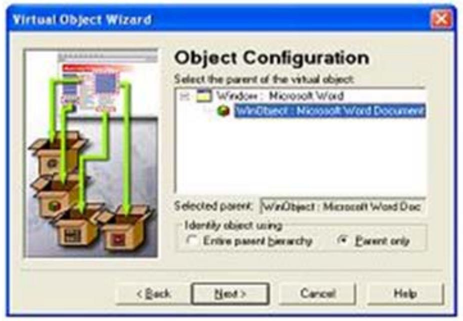
- Currently only supported in QTP.



Mark Virtual Object

1. Click the Mark Object button. (This screen is minimized.)
2. Use the crosshairs pointer to outline the virtual object.
   Do not include scroll bars or labels in the virtual object.

Width: 61
Height: 71



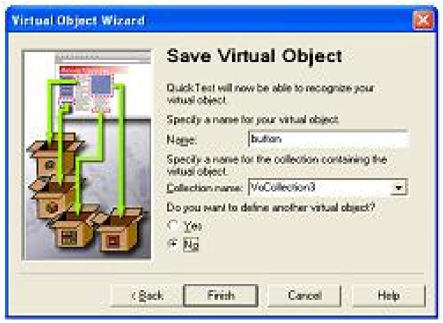
Object Configuration

Select the parent of the virtual object:

Window : Microsoft Word
WinObject : Microsoft Word Document

Selected parent: WinObject : Microsoft Word Doc

Identify object using
Entire parent hierarchy [✓ Parent only]



Save Virtual Object

QuickTest will now be able to recognize your virtual object.

Specify a name for your virtual object.

Name: button

Specify a name for the collection containing the virtual object.

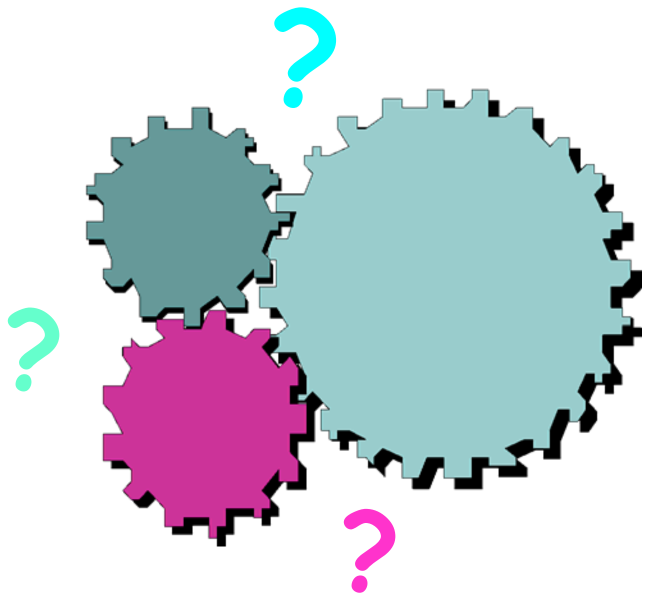
Collection name: VoCollection3

Do you want to define another virtual object?

Yes [✓ No]

Issue 2: Test Oracle

- How to check the expected program output?
  - Sums, totals, balances
  - Data displays
  - Error messages
  - Cursor movement
  - Window handling
  - Queries



- How to check the expected output consistently from build to build?

Checkpoint

- A **checkpoint** is a point in a script that you insert to check the state of the system
  - Called *checkpoint* in QTP and TestComplete
  - Called *verification point* in RFT and Marathon
- A checkpoint serves as your eyes: record a verification point wherever you need to verify expected results
  - The insertion of checkpoint depends on GUI object identification: first specify a GUI object, then specify its expected value.

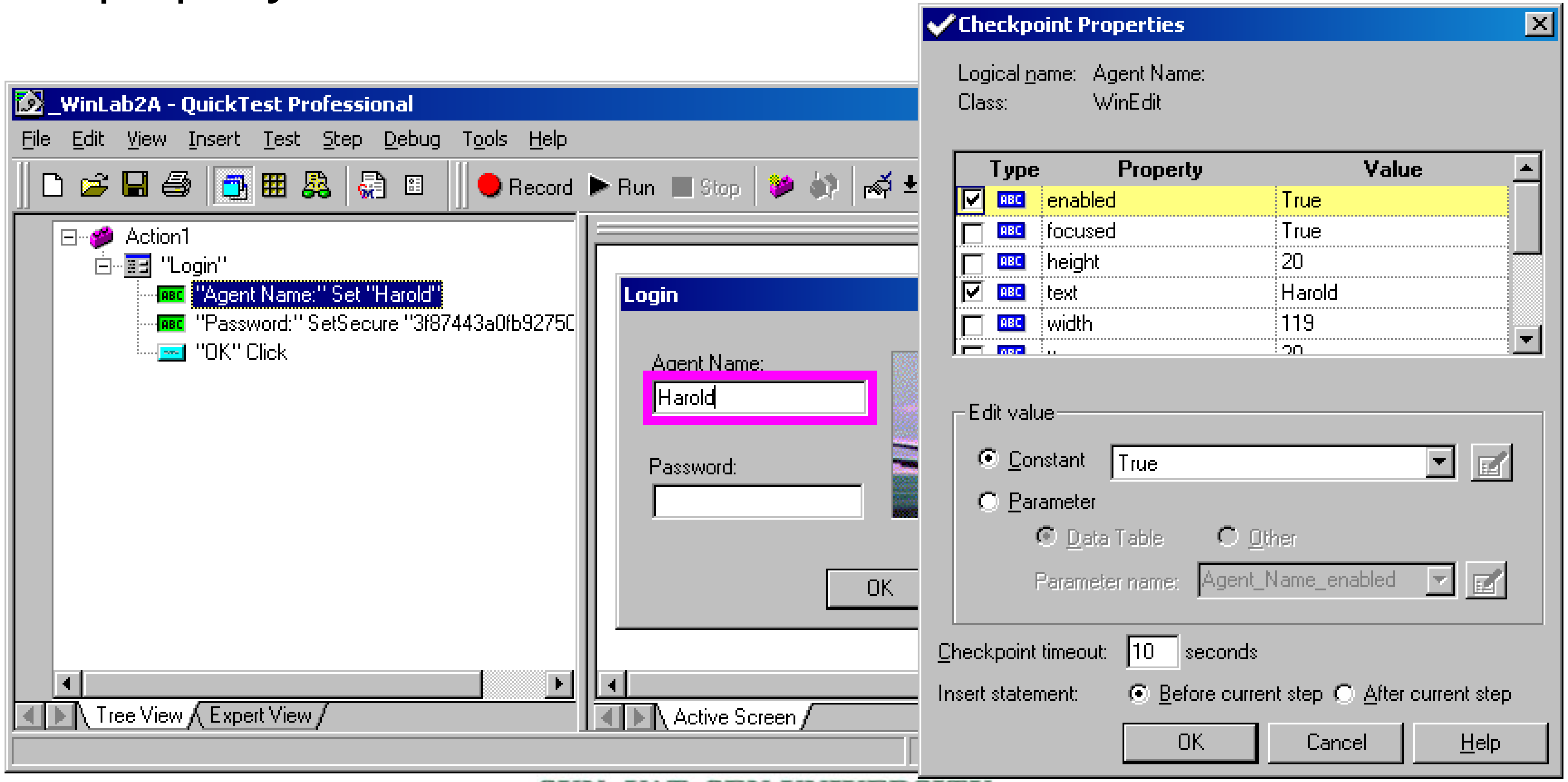
Common Types of Checkpoint

- Property Checkpoint
- Image Checkpoint
- Table Checkpoint
- Database Checkpoint
- …

- We will use QTP as an example to introduce them.

Property Checkpoint

- Check whether the properties of a GUI object are equal to specific values.
  - Sometimes we need to use regular expressions to check the property values.


_WinLab2A - QuickTest Professional
File Edit View Insert Test Step Debug Tools Help
Record Run Stop
Tree View Expert View Active Screen
Login
OK

Action1
"Login"
"Agent Name:" Set "Harold"
"Password:" SetSecure "3f87443a0fb9275
"OK" Click

Agent Name: Harold
Password:

Checkpoint Properties

Logical name: Agent Name:
Class: WinEdit

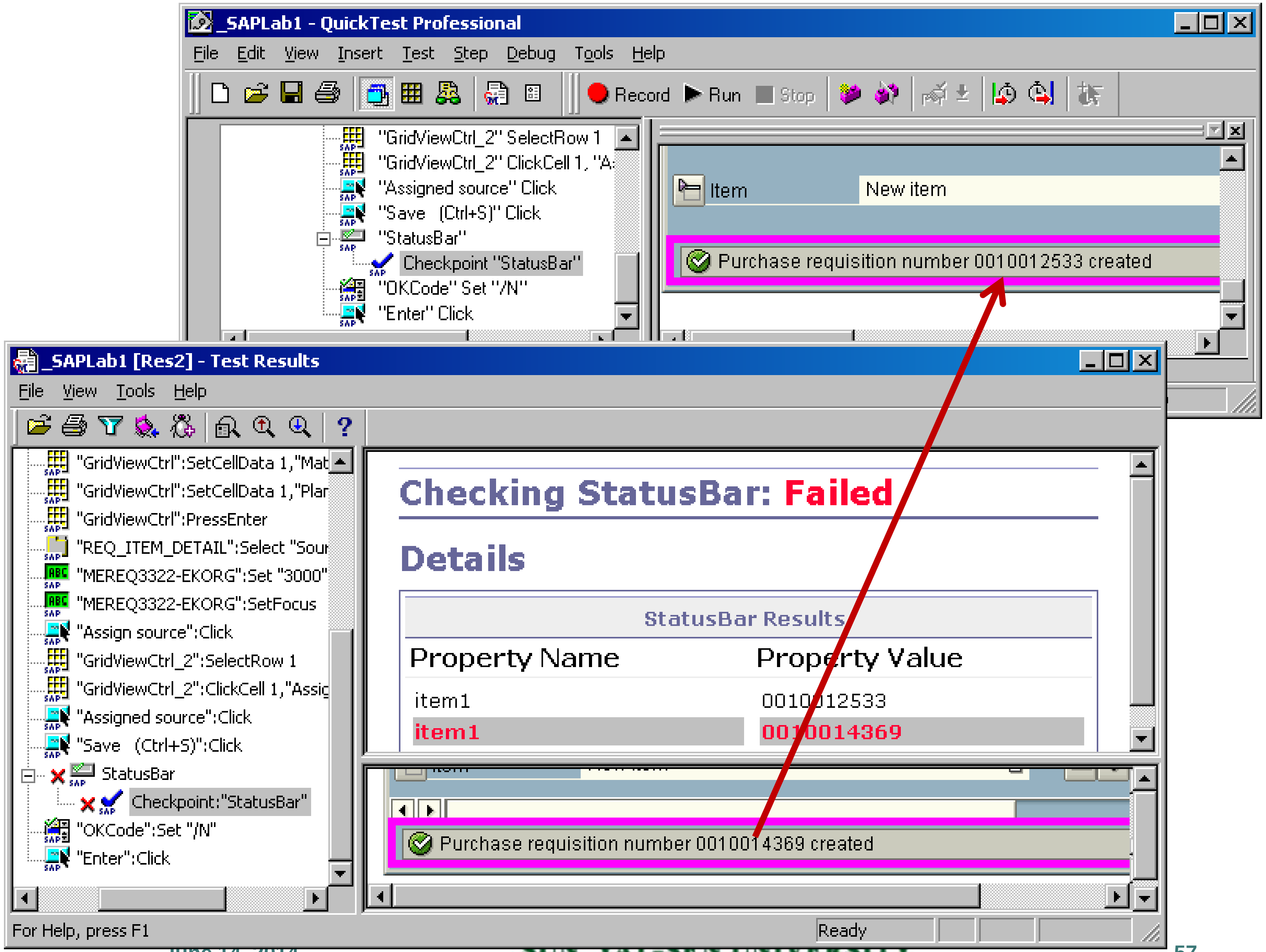
	Type	Property	Value
✓	ABC	enabled	True
	ABC	focused	True
	ABC	height	20
✓	ABC	text	Harold
	ABC	width	119

Edit value
[✓ Constant] True
Parameter
[✓ Data Table] Other
Parameter name: Agent_Name_enabled

Checkpoint timeout: 10 seconds
Insert statement: [✓ Before current step] After current step

OK Cancel Help

Where Regular Expression is Needed


_SAPLab1 - QuickTest Professional
File Edit View Insert Test Step Debug Tools Help
Record Run Stop
"GridViewCtrl_2" SelectRow 1
"GridViewCtrl_2" ClickCell 1, "A
"Assigned source" Click
"Save (Ctrl+S)" Click
"StatusBar"
Checkpoint "StatusBar"
"OKCode" Set "/N"
"Enter" Click
Item
New item
_SAPLab1 [Res2] - Test Results
File View Tools Help
"GridViewCtrl":SetCellData 1,"Mat
"GridViewCtrl":SetCellData 1,"Plar
"GridViewCtrl":PressEnter
"REQ_ITEM_DETAIL":Select "Sour
"MEREQ3322-EKORG":Set "3000"
"MEREQ3322-EKORG":SetFocus
"Assign source":Click
"GridViewCtrl_2":SelectRow 1
"GridViewCtrl_2":ClickCell 1,"Assig
"Assigned source":Click
"Save (Ctrl+S)":Click
StatusBar
Checkpoint:"StatusBar"
"OKCode":Set "/N"
"Enter":Click
For Help, press F1
Ready

Purchase requisition number 0010012533 created

Checking StatusBar: Failed

Details

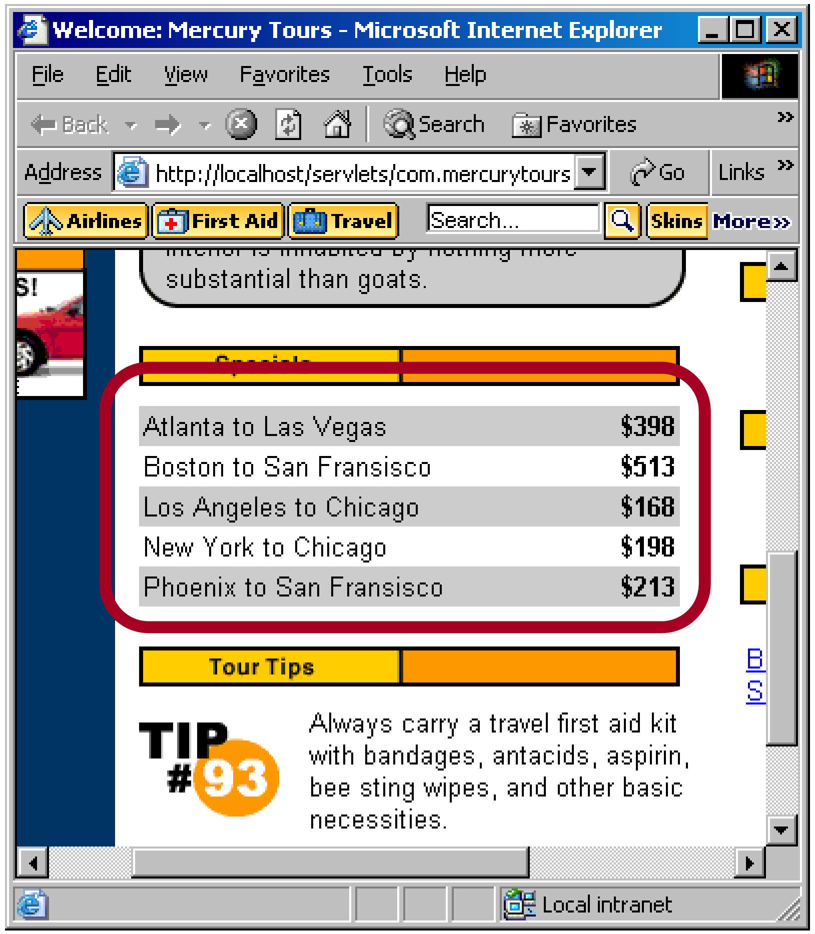
StatusBar Results

Property Name	Property Value
item1	0010012533
item1	0010014369

Purchase requisition number 0010014369 created

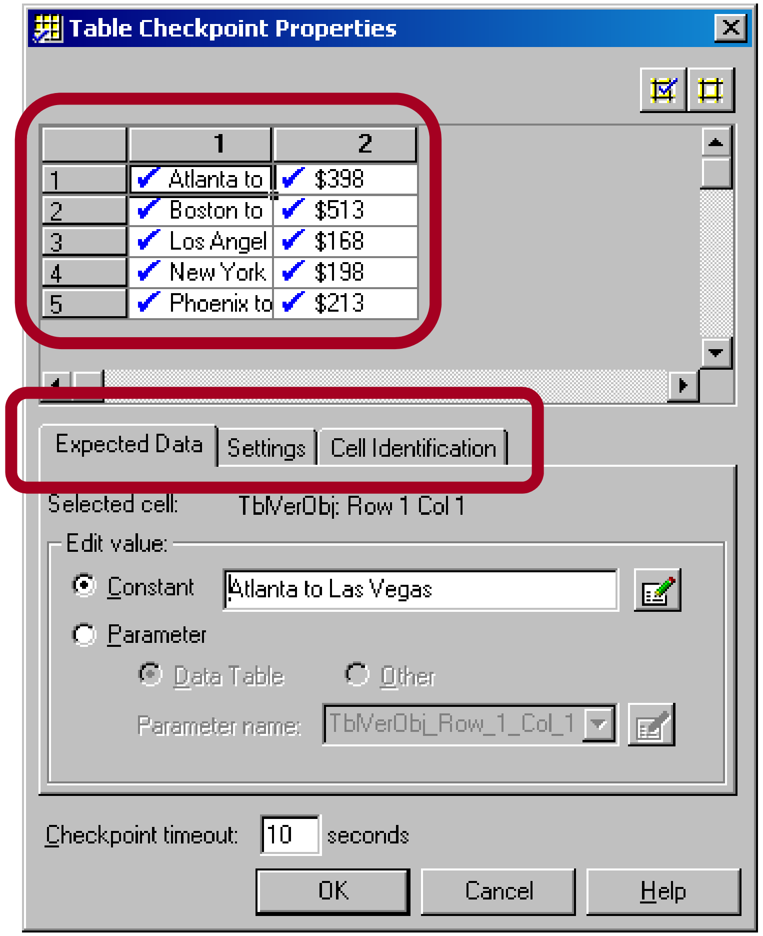
Table Checkpoint

A table checkpoint can check the values of cells in an application.


Welcome: Mercury Tours - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost/servlets/com.mercurytours Go Links
Airlines First Aid Travel Search... Skins More»
substantial than goats.
Specials
Tour Tips
TIP #93
Local intranet

Atlanta to Las Vegas	$398
Boston to San Fransisco	$513
Los Angeles to Chicago	$168
New York to Chicago	$198
Phoenix to San Fransisco	$213

Always carry a travel first aid kit with bandages, antacids, aspirin, bee sting wipes, and other basic necessities.



	1	2
1	✓ Atlanta to	✓ $398
2	✓ Boston to	✓ $513
3	✓ Los Angel	✓ $168
4	✓ New York	✓ $198
5	✓ Phoenix to	✓ $213

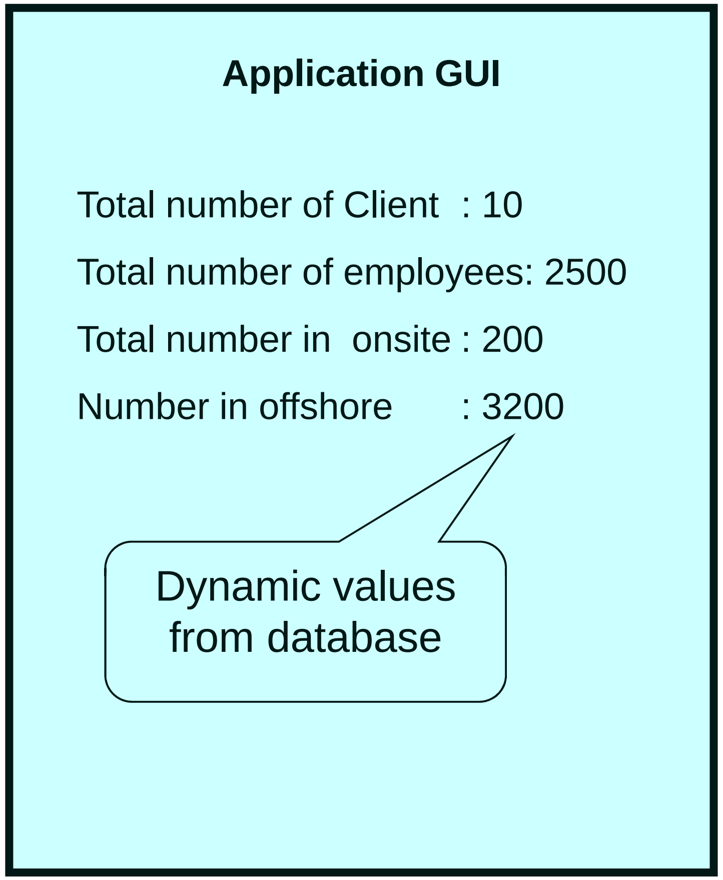
Selected cell: TblVerObj: Row 1 Col 1

Edit value:
[✓ Constant] Atlanta to Las Vegas
Parameter
[✓ Data Table] Other
Parameter name: TblVerObj_Row_1_Col_1

Checkpoint timeout: 10 seconds

Database Check Points

Compares the dynamic values displayed in the application with the values from the database. User will write a SQL query when creating this checkpoint with which QTP will check the dynamic values displayed in application.


Application GUI
Total number of Client : 10
Total number of employees: 2500
Total number in onsite : 200
Number in offshore : 3200
Dynamic values from database

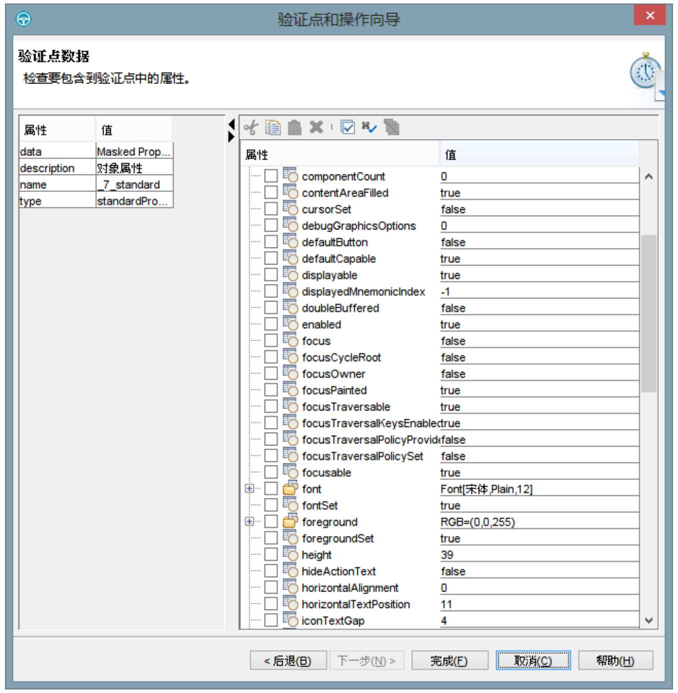
Query written in QTP to check values:
Select * from data where company = 'Software Testing';

Values retrieved from database by QTP

Cl_num	emp_num	on_num	off_num
10	2500	200	3200

Tool Implementation

- RFT support property checkpoints, data checkpoints (a special case of property checkpoints), and image checkpoints.
  - It also support customized checkpoint by extracting the values of properties into Java variables.

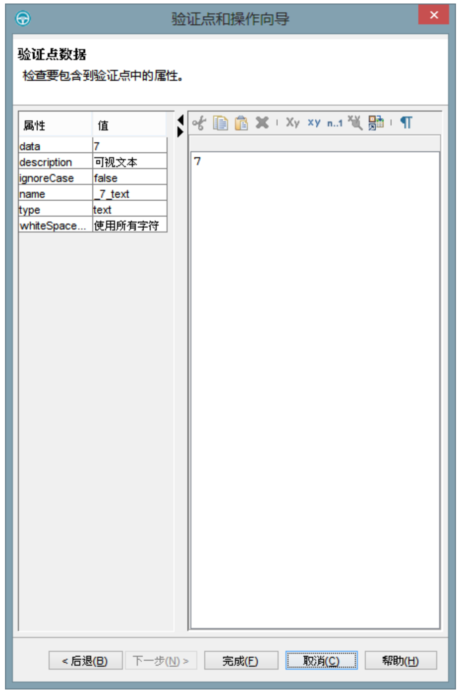


验证点数据
检查要包含到验证点中的属性。

属性	值
data	Masked Prop...
description	对象属性
name	_7_standard
type	standardPro...

属性	值
componentCount	0
contentAreaFilled	true
cursorSet	false
debugGraphicsOptions	0
defaultButton	false
defaultCapable	true
displayable	true
displayedMnemonicIndex	-1
doubleBuffered	false
enabled	true
focus	false
focusCycleRoot	false
focusOwner	false
focusPainted	true
focusTraversable	true
focusTraversalKeysEnabled	true
focusTraversalPolicyProvid	false
focusTraversalPolicySet	false
focusable	true
font	Font[宋体,Plain,12]
fontSet	true
foreground	RGB=(0,0,255)
foregroundSet	true
height	39
hideActionText	false
horizontalAlignment	0
horizontalTextPosition	11
iconTextGap	4

property checkpoints

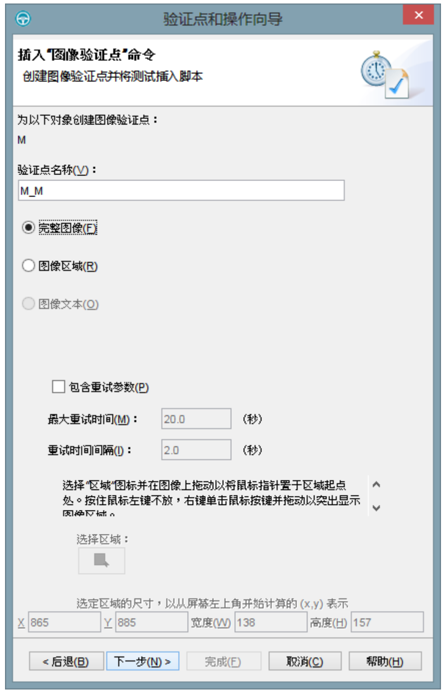


验证点数据
检查要包含到验证点中的属性。

属性	值
data	7
description	可视文本
ignoreCase	false
name	_7_text
type	text
whiteSpace...	使用所有字符

7

data checkpoints



插入"图像验证点"命令
创建图像验证点并将测试插入脚本

为以下对象创建图像验证点：
M

验证点名称(V)：
M_M

[✓ 完整图像(F)]
图像区域(R)
图像文本(O)

☐ 包含重试参数(P)
最大重试时间(M)： 20.0 (秒)
重试时间间隔(I)： 2.0 (秒)

选择"区域"图标并在图像上拖动以将鼠标指针置于区域起点处。按住鼠标左键不放，右键单击鼠标按键并拖动以突出显示图像区域。

选择区域：

选定区域的尺寸，以从屏幕左上角开始计算的 (x,y) 表示
X 865 Y 885 宽度(W) 138 高度(H) 157

image checkpoints

Cons and Pros of Capture/Replay

- Pros:
  - Conceptually simple.
    - Dominant script development paradigm
  - Straightforward.
    - Don’t need coding.
    - Don’t need to understand test script language.
- Cons:
  - Low fault detection capability
    - Rerunning old tests that the program has passed is less powerful than running new tests.
    - Old tests do not address new features.
  - Interface changes force maintenance of tests.
    - Most early test failures are due to interface changes.
  - Cannot apply until the product is stabilized.
    - Too late to find bug!
    - Cannot get timely feedback from users.

Test Automation Techniques

- Capture/replay
- Structural Test Scripts Development
- Data-driven test automation
- Keyword-driven test automation

- Mainstream Tools:
  - HP QuickTest Professional
  - IBM Rational Functional Test
  - AutomatedQA TestComplete
  - Borland SilkTest
  - Marathon

The idea

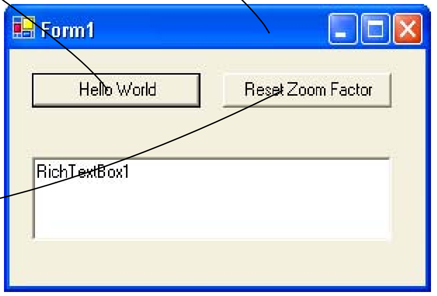
- Test script development with structured programming constructs.
  - Control statement: conditional/switch statement, loop statement
  - Modular design: functional call, library
  - Exception handling
- Recall: test automation is software development
  - Encapsulation and Reuse

Example: No Modular Design

Actual Recorded Script:

```
Option Explicit

Sub Main()
' Begin Record on Friday, April 14, 2006 at 4:41:34 PM by veragep

  ' Attach to Application=EXPLORER.EXE Caption='F:\SQuAD Presentation on D'
  Window("Application=EXPLORER.EXE Caption='F:\SQuAD Presentation on D'").Attach
    ListView("Index=1").Select "TinyApp.exe"
    ListView("Index=1").Select "TinyApp.exe", tpMouseDoubleClick

  ' Attach to Name=Form1 Application=TINYAPP.EXE
  DotNETForm("Name=Form1 Application=TINYAPP.EXE").Attach
    DotNETForm.Size 304, 192
    DotNETForm.Move 22, 29
    DotNETButton("Name=Button1").DoubleClick
    DotNETButton("Name=Button2").Click
    DotNETForm.Close

' End Record on Friday, April 14, 2006 at 4:41:46 PM by veragep

End Sub
```

Test script recorded by the capture/replay tool = A big “main” function

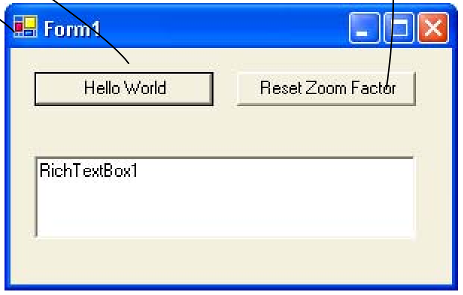


Example: Modular Design

```
Option Explicit

Sub Main()
   StartApp "G:\SQuAD Presentation\Tiny App\bin\TinyApp.exe"
   Button1Click
   Button2Click
End Sub

Sub StartApp(strFullAppName As String)
   Shell strFullAppName, vbNormalFocus
End Sub
Sub Button1Click()
   Form1Attach
   DotNETButton("Name=Button1").Click
End Sub
Sub Button2Click()
   Form1Attach
   DotNETButton("Name=Button2").Click
End Sub
Sub Form1Attach()
   DotNETForm("Name=Form1 Application=TINYAPP.EXE").Attach
End Sub
```

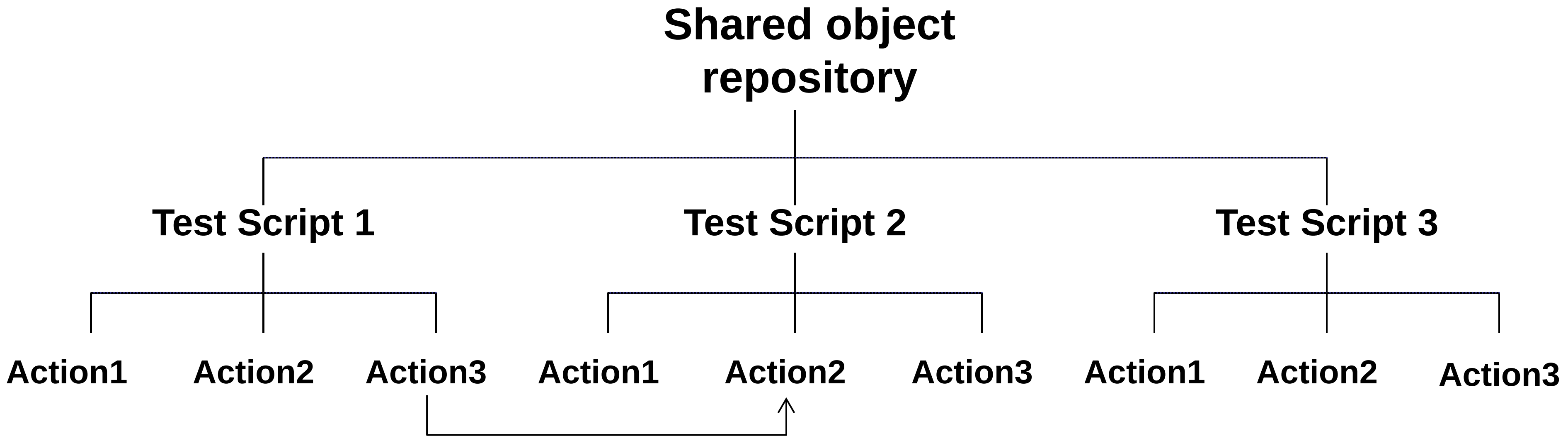
Test script refactored to have a better design

$$\begin{tikzcd}
 & \text{main} \arrow[dl] \arrow[d] \arrow[dr] & \\
\text{StartApp} & \text{Button1Click} \arrow[dr] & \text{Button2Click} \arrow[d] \\
 & & \text{Form1Attach}
\end{tikzcd}$$



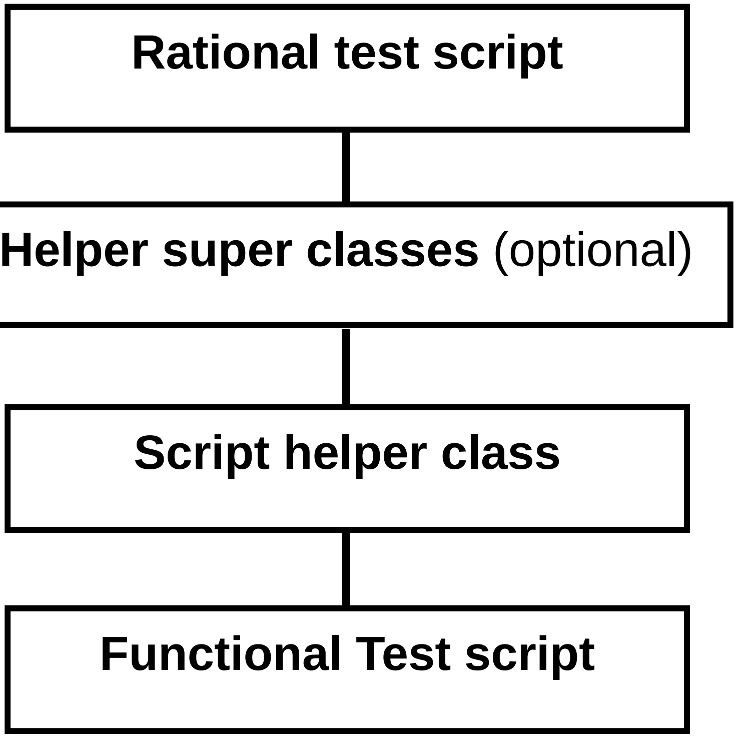
Script Reuse in QTP

- QTP test scripts are organized by **actions**
  - A test script consists of one or more actions
  - Actions can be invoked multiple times within a test script
  - Actions can be marked as reusable so they can be called from other tests
    - Only reusable actions can be called from other tests
    - Such reusable actions can also have parameters like functions



Script Reuse in RFT

- RFT test scripts are organized by **script classes**
  - Different test scripts can share a common helper super classes


Rational test script
Helper super classes (optional)
Script helper class
Functional Test script

Test Automation Techniques

- Capture/replay
- Structural Test Scripts Development
- Data-driven test automation
- Keyword-driven test automation

- Mainstream Tools:
  - HP QuickTest Professional
  - IBM Rational Functional Test
  - AutomatedQA TestComplete
  - Borland SilkTest
  - Marathon

The idea

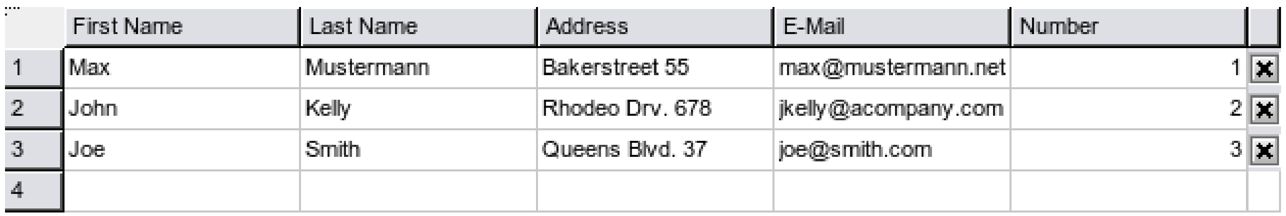
- Data-focused automation. Data is defined in external data source and de-coupled from test script.

- Users define the data sets, while the test engineer develop the test scripts to load the data sets.

- Good fit for testing that features multi-environment, big datasets, and rarely changing test scripts.
  - Test the boundaries of a quantity field in an online retail application
  - Test variable length data in a text input field
  - Test the order-total functionality of a retail cash register

Example

Data sets: defined by the users



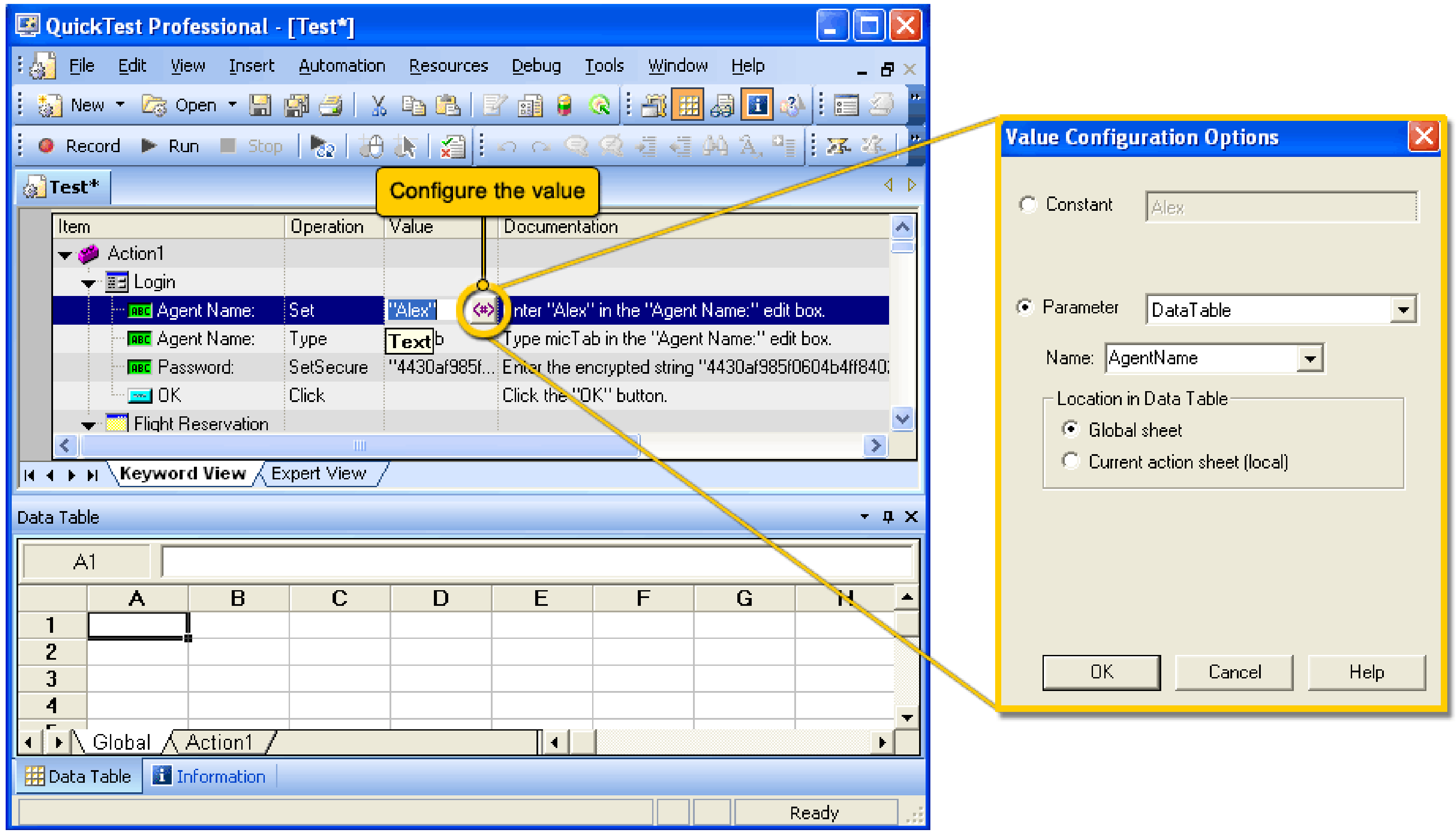
	First Name	Last Name	Address	E-Mail	Number	
1	Max	Mustermann	Bakerstreet 55	max@mustermann.net	1	☒
2	John	Kelly	Rhodeo Drv. 678	jkelly@acompany.com	2	☒
3	Joe	Smith	Queens Blvd. 37	joe@smith.com	3	☒
4						

Test scripts: developed by the test engineer

```
for address_record in testData.dataset("addresses.tsv"):
    firstName = testData.field(address_record, "First Name")
    lastName = testData.field(address_record, "Last Name")
    address = testData.field(address_record, "Address")
    email = testData.field(address_record, "E-Mail")
```

Data-driven Test Automation in QTP

The reference to a property in a test step or a verification point can be parameterized by a column name in a data set. During replay, values of such properties are fetched from the data set. One row for one test run.



Item	Operation	Value	Documentation
Action1			
Login			
Agent Name:	Set	"Alex"	Enter "Alex" in the "Agent Name:" edit box.
Agent Name:	Type	Text	Type micTab in the "Agent Name:" edit box.
Password:	SetSecure	"4430af985f...	Enter the encrypted string "4430af985f0604b4ff840
OK	Click		Click the "OK" button.
Flight Reservation			

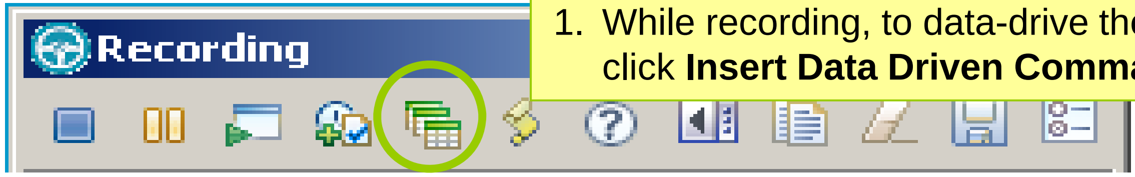
A1

	A	B	C	D	E	F	G	H
1								
2								
3								
4								

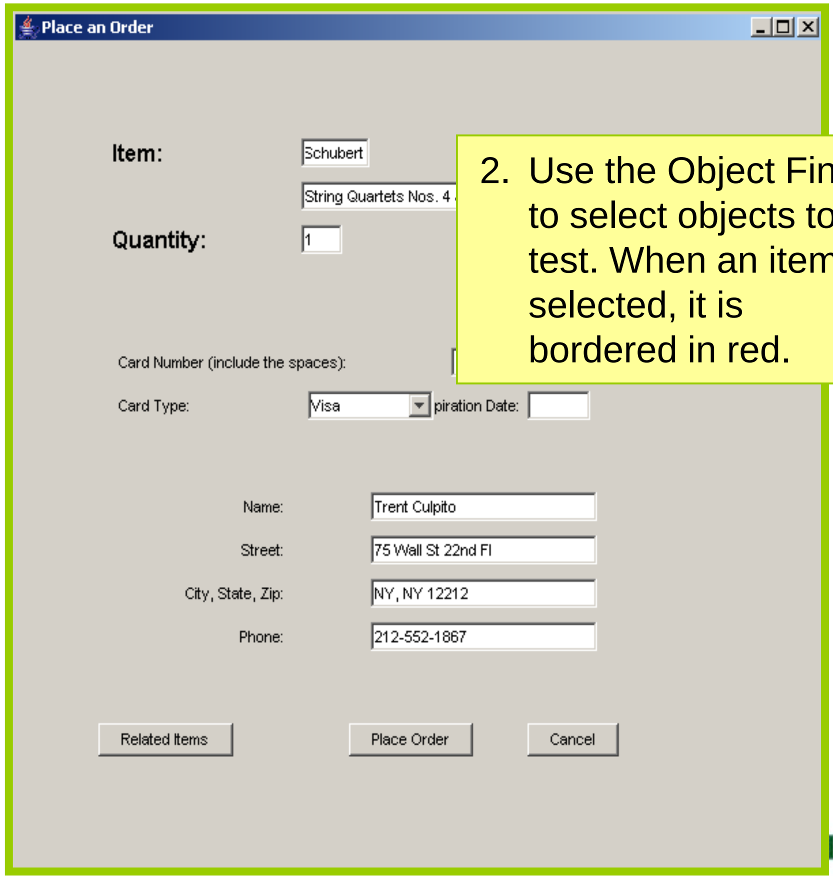
Constant: Alex
[✓ Parameter]: DataTable
Name: AgentName
Location in Data Table: [✓ Global sheet] Current action sheet (local)

Data-driven Test Automation in RFT

Similar to QTP, except that RFT additional supports the creation of data set schema from GUI object properties.

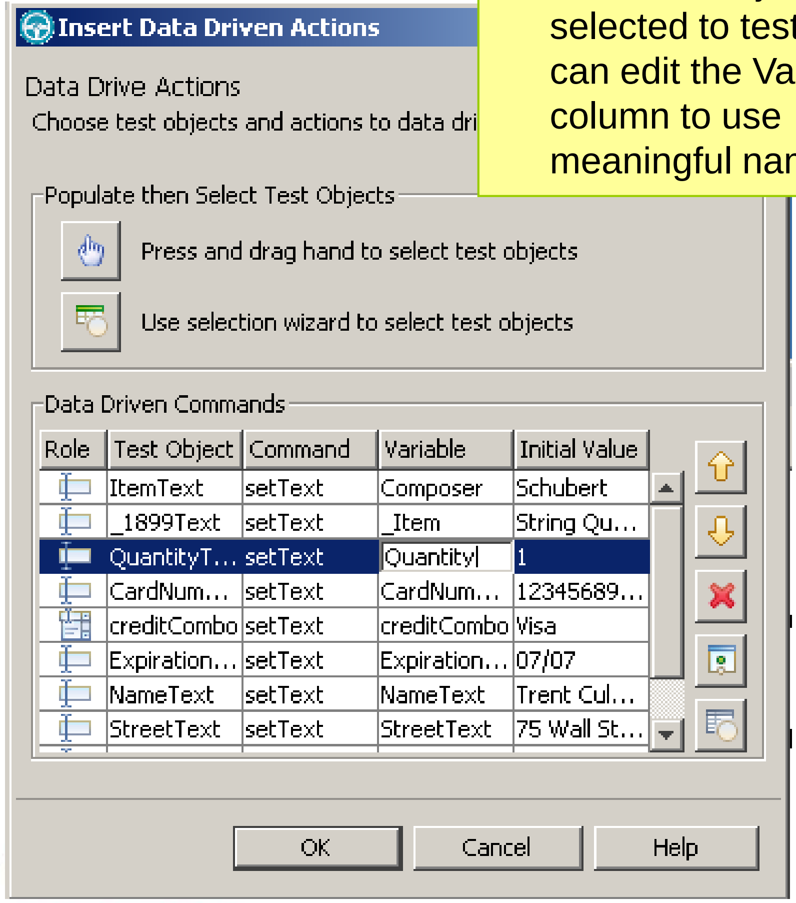


1. While recording, to data-drive the script, click **Insert Data Driven Commands**.



Item:	Schubert
	String Quartets Nos. 4
Quantity:	1
Card Number (include the spaces):	
Card Type:	Visa
Expiration Date:	
Name:	Trent Culpito
Street:	75 Wall St 22nd Fl
City, State, Zip:	NY, NY 12212
Phone:	212-552-1867

2. Use the Object Finder to select objects to test. When an item is selected, it is bordered in red.



Data Drive Actions

Choose test objects and actions to data dri

Populate then Select Test Objects

Press and drag hand to select test objects

Use selection wizard to select test objects

Data Driven Commands

Role	Test Object	Command	Variable	Initial Value
	ItemText	setText	Composer	Schubert
	_1899Text	setText	_Item	String Qu...
	QuantityT...	setText	Quantity	1
	CardNum...	setText	CardNum...	12345689...
	creditCombo	setText	creditCombo	Visa
	Expiration...	setText	Expiration...	07/07
	NameText	setText	NameText	Trent Cul...
	StreetText	setText	StreetText	75 Wall St...

3. RFT collects data about the object you selected to test. You can edit the Variable column to use meaningful names.

Test Automation Techniques

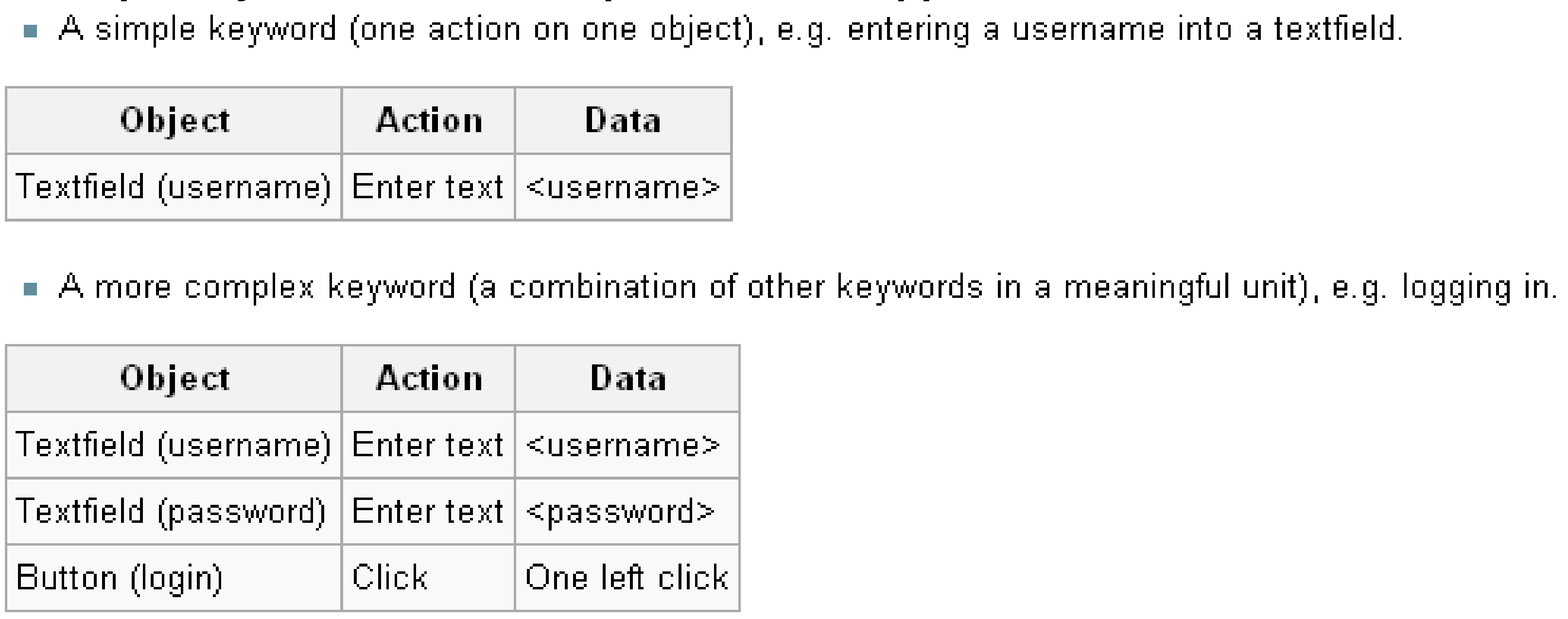
- Capture/replay
- Structural Test Scripts Development
- Data-driven test automation
- Keyword-driven test automation

- Mainstream Tools:
  - HP QuickTest Professional
  - IBM Rational Functional Test
  - AutomatedQA TestComplete
  - Borland SilkTest
  - Marathon

The idea

- It is also known as action-word testing
- Center around the concept of **keywords**
  - Keywords are implemented by ***test engineers***.
  - Test scripts are composed by ***domain experts*** using keywords.

Example keywords for a web-questionnaire application:



- A simple keyword (one action on one object), e.g. entering a username into a textfield.

Object	Action	Data
Textfield (username)	Enter text	<username>

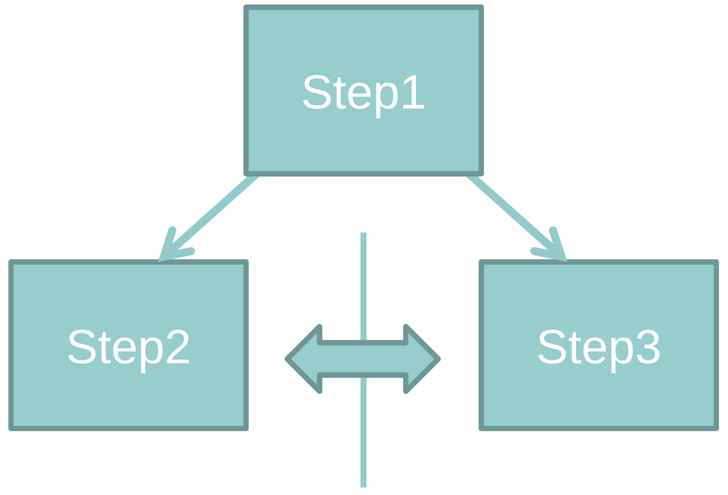
- A more complex keyword (a combination of other keywords in a meaningful unit), e.g. logging in.

Object	Action	Data
Textfield (username)	Enter text	<username>
Textfield (password)	Enter text	<password>
Button (login)	Click	One left click

Keyword-Driven Test Automation

- Usually involves three steps:
  - **Step1**: Domain experts and test engineers work together to identify operations that can be modeled as keywords.
  - **Step2**: Test engineers implement the keywords.
  - **Step3**: Domain experts design test cases with the keywords.

Test Engineer



Keyword interfaces

Domain Expert

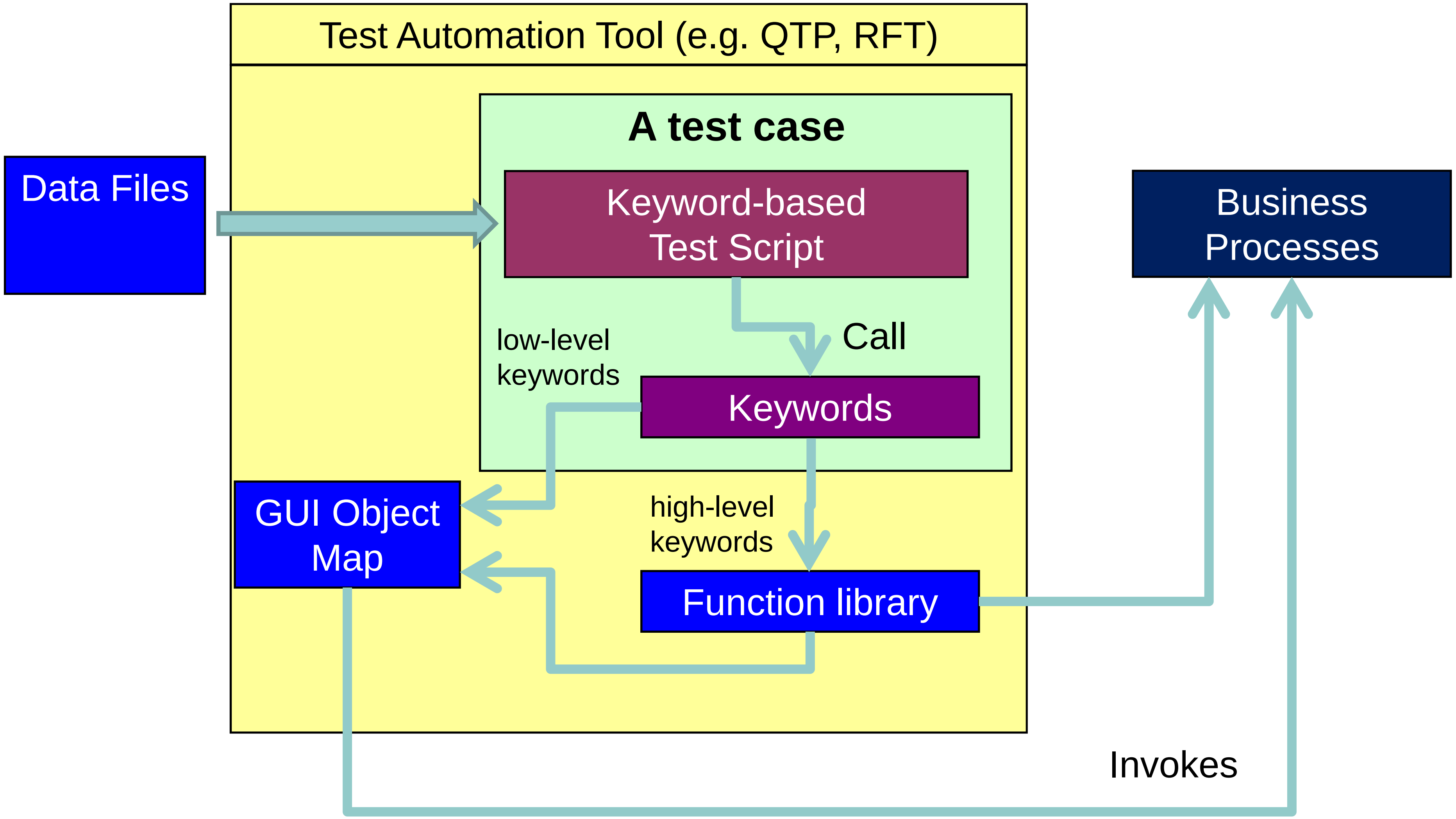
Key Benefits

- Better utilization of expertise.
- Domain experts as end-users of the software get involved in testing earlier (and find problems earlier).
- Test case design activity can be started earlier (dos not need to wait until the GUI is stabilized).
- Test scripts are much easier to understand and maintain.

Types of Keyword

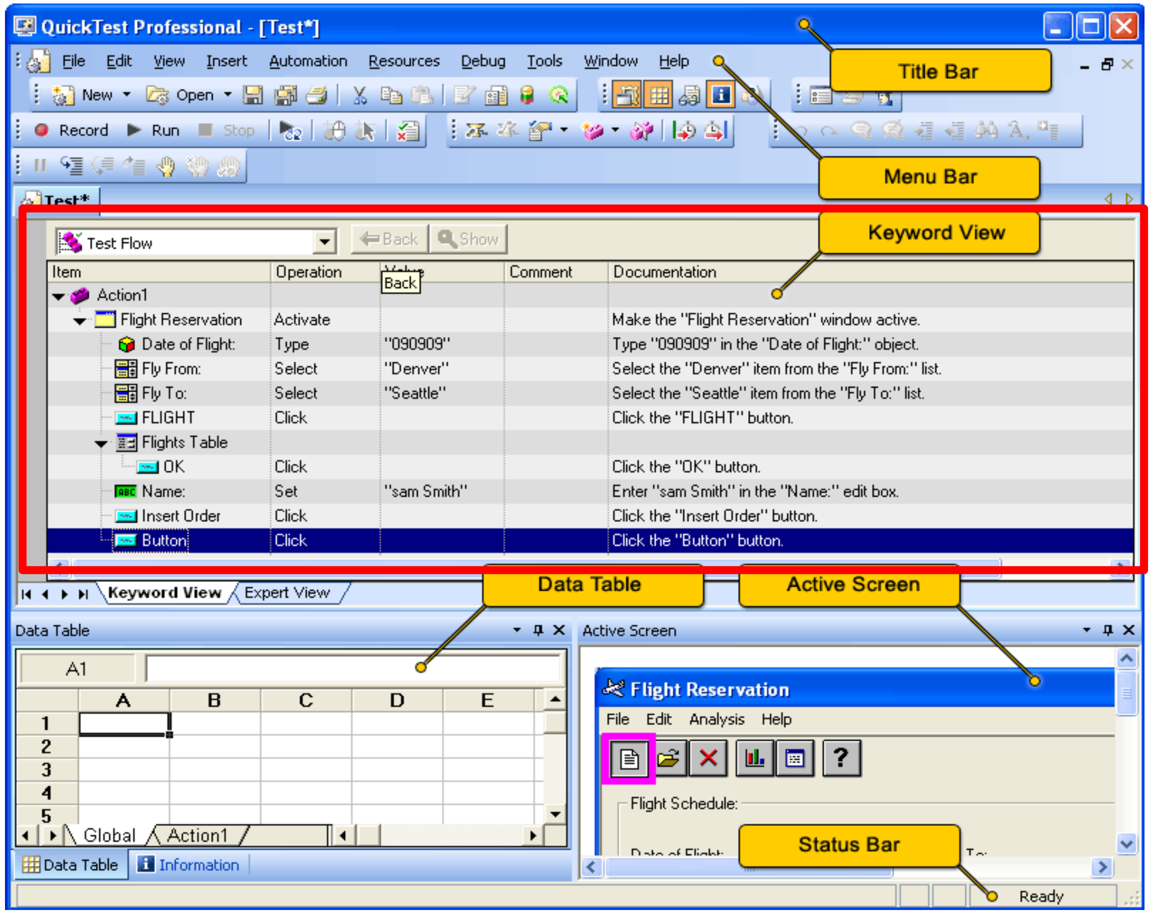
- Low-level keywords
  - Keyword functions are implemented as **operations on GUI objects**. (e.g. button click, combo list selection)
  - Defined by the object map automatically. No efforts required from the test engineers.
  - Deeply coupled with the GUI. Tedious to use. Not available before the GUI is stabilized.
- High-level keywords
  - Keyword functions are implemented as **functions in script languages** (e.g. log–in, check-out).
  - Defined by test engineers. Significant efforts required.
  - Function interfaces provide a level of abstraction. Domain experts can start their work earlier.

Framework


Test Automation Tool (e.g. QTP, RFT)
A test case
Data Files
Keyword-based Test Script
Business Processes
Call
low-level keywords
Keywords
GUI Object Map
high-level keywords
Function library
Invokes

Keyword-driven in QTP

- Low-level keywords are defined with the object map automatically.
- They can be inserted into the test script in the keyword view.



Item	Operation	Value	Comment	Documentation
Action1				
Flight Reservation	Activate			Make the "Flight Reservation" window active.
Date of Flight:	Type	"090909"		Type "090909" in the "Date of Flight:" object.
Fly From:	Select	"Denver"		Select the "Denver" item from the "Fly From:" list.
Fly To:	Select	"Seattle"		Select the "Seattle" item from the "Fly To:" list.
FLIGHT	Click			Click the "FLIGHT" button.
Flights Table				
OK	Click			Click the "OK" button.
Name:	Set	"sam Smith"		Enter "sam Smith" in the "Name:" edit box.
Insert Order	Click			Click the "Insert Order" button.
Button	Click			Click the "Button" button.

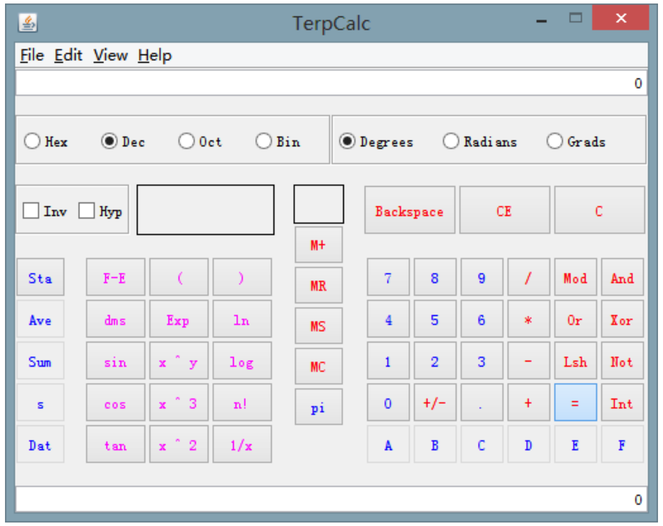
Keyword-driven in QTP

- High-level keywords are implemented as VBScript functions.
- Example:

```
Function VerifyButtonStatus(ButtonName, ExpectedStatus)

    ActualStatus = wlndow("Flight Reservation").
             WinButton(ButtonName).GetROProperty("enabled')

    If ActuaStatus = ExpectedStatus Then
        Reporter.ReportEvent(micpass,"Verify Button",
            “Button status was as expected.“)
    else
        Reporter.ReportEvent(micFail,"Verify Button",
            “Button status was unexpected.“)
    End If
End Function

Function Call: VerifyButtonStatus("FLIGHT", False)
```

Case Study: TerpCalc

- Opportunity for data-driven test automation
  - **Table**: left operand, operator, right operand, expected result
  - **Table**: decimal, expected hexadecimal, expected octal, expected binary
  - **Table**: values, statistics function, expected value
  - …
- Opportunity for keyword-driven test automation
  - **Keywords**: enter a list of values, enter a number, …



Maintenance of Test Cases

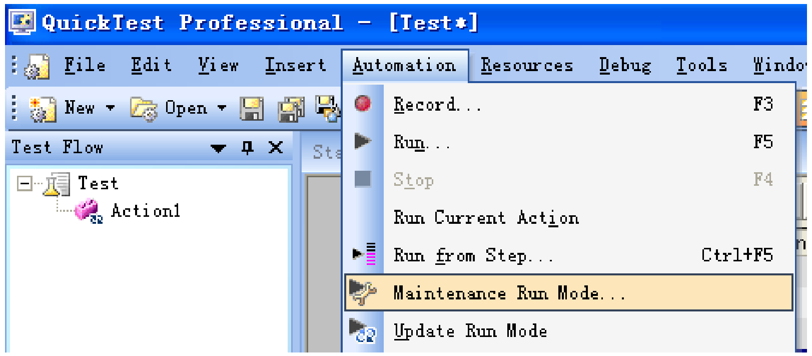
- Code maintenance is expensive and inevitable.
  - Ditto to test cases
- When test cases are executed manually, the effort is mainly spent on maintain the documents.
- When the execution of test cases are automated, additional effort needs to be spent on maintaining test scripts, data table, function library, and object map.

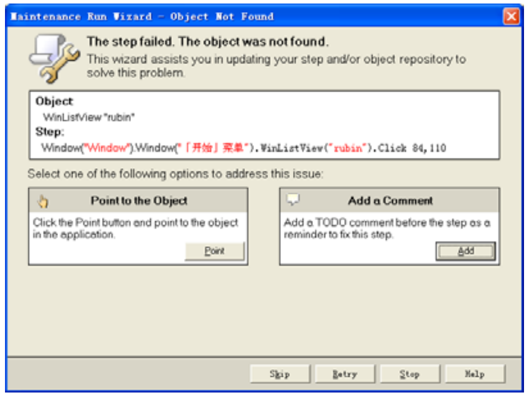
Types of Maintenance

- Corrective maintenance: keeping changes in system functionality co-ordinated with the test suite
  - Changes in UI
  - Changes in business process
  - Changes in data dependencies
  - Changes in data format (storage, presentation, internal)
- Preventive maintenance: test suite "clean-up"
  - Removing tests that are no longer relevant.
  - Accumulation of duplicate / redundant tests.
  - Removal or refactoring of unreliable tests.

Test Script Maintenance in QTP and RFT

- Both support test script debugging (breakpoints, watch list, …)
- QTP supports "Maintenance run" and "Update run" mode




Maintenance Run Wizard - Object Not Found

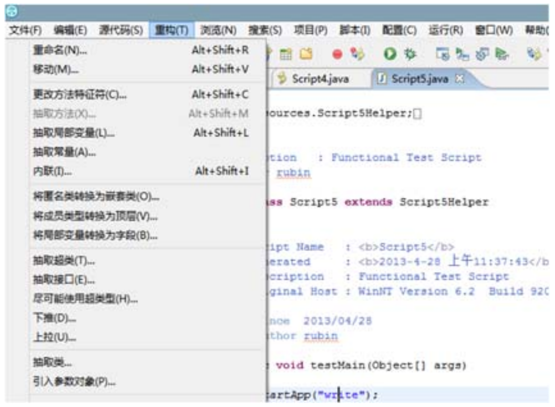
The step failed. The object was not found.
This wizard assists you in updating your step and/or object repository to solve this problem.

Object
WinListView "rubin"
Step:
Window("Window").Window("「开始」菜单").WinListView("rubin").Click 84,110

Select one of the following options to address this issue:

Point to the Object
Click the Point button and point to the object in the application.
Point
Add a Comment
Add a TODO comment before the step as a reminder to fix this step.
Add
Skip
Retry
Stop
Help

- RFT supports test script refactory



```
ources.Script5Helper;

tion   : Functional Test Script
 rubin

ss Script5 extends Script5Helper

ipt Name    : <b>Script5</b>
erated      : <b>2013-4-28 上午11:37:43</b
cription    : Functional Test Script
ginal Host  : WinNT Version 6.2  Build 92C

nce  2013/04/28
thor rubin

 void testMain(Object[] args)

artApp("write");
```

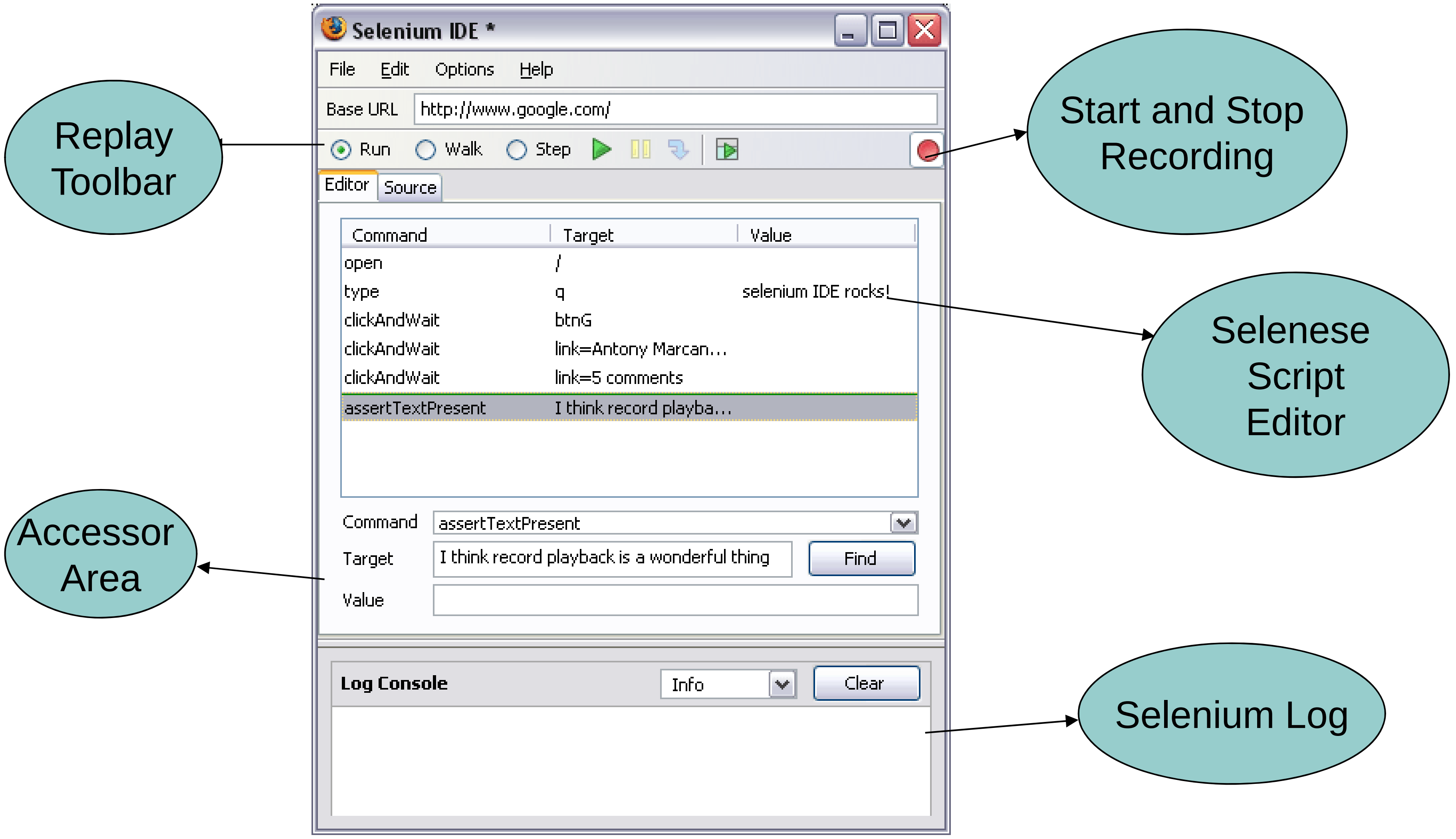
Test Automation for Web: Selenium

- Selenium is a robust set of tools that supports rapid development of test automation for web-based applications.
- Selenium provides a rich set of testing functions specifically geared to the needs of testing of a web application.
- Selenium operations are highly flexible, allowing many options for locating UI elements and comparing expected test results against actual application behavior.

Selenium Features

- Supports Cross Browser Testing. The Selenium tests can be run on multiple browsers.
- Allows scripting in several languages like Java, C#, PHP and Python.
- Assertion statements provide an efficient way of comparing expected and actual results.
- Inbuilt reporting mechanism.

Capture & Replay with Selenium IDE



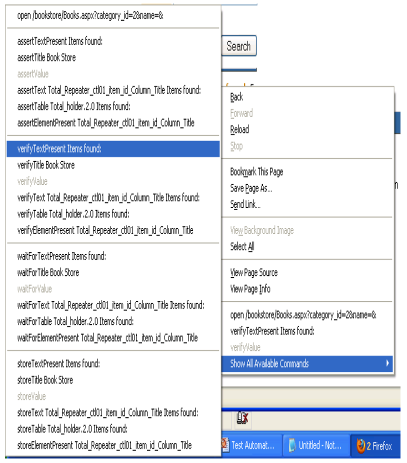
Command	Target	Value
open	/	
type	q	selenium IDE rocks!
clickAndWait	btnG	
clickAndWait	link=Antony Marcan...	
clickAndWait	link=5 comments	
assertTextPresent	I think record playba...	

Command: assertTextPresent
Target: I think record playback is a wonderful thing
Value:

Adding Checkpoints

- Two kinds of checkpoitns: verifications and assertions. They are used to check if
  - an element is present somewhere on the page?
  - specific text is somewhere on the page?
  - specific text is at a specific location on the page?

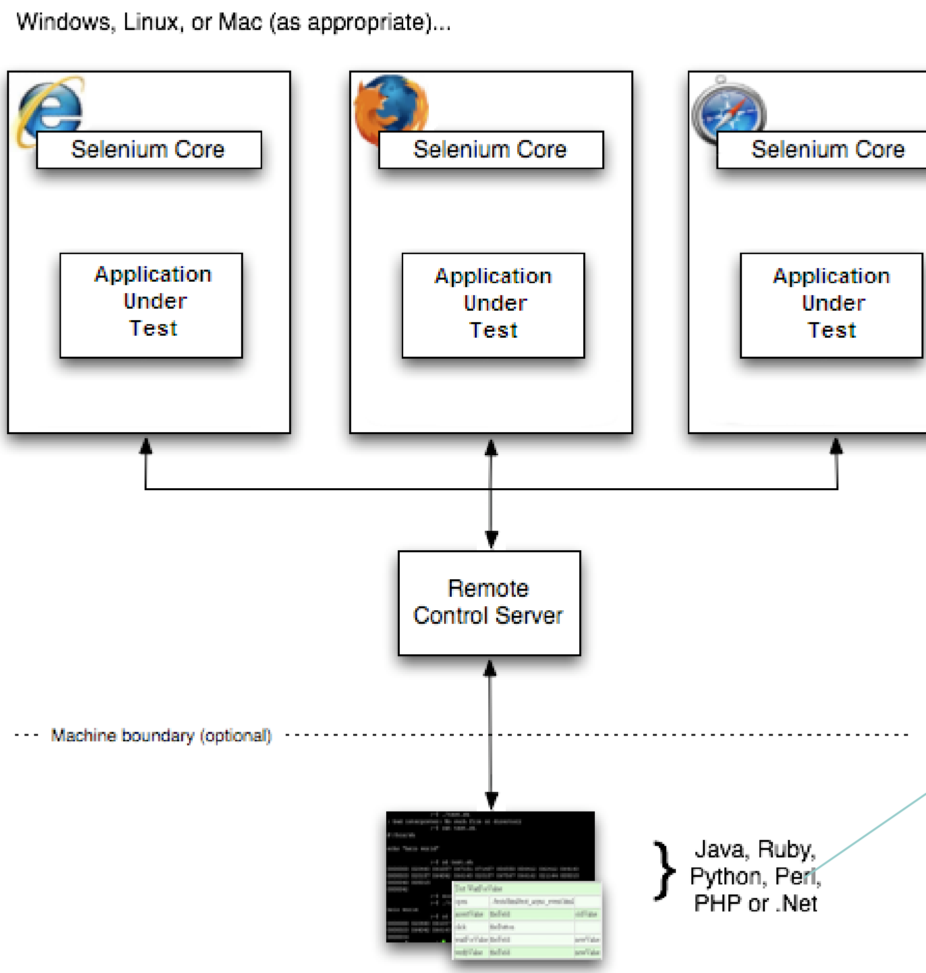
- The difference between verifications and assertions:
  - If an assertion fails, the script will be aborted but if a verification fails the script will continue.

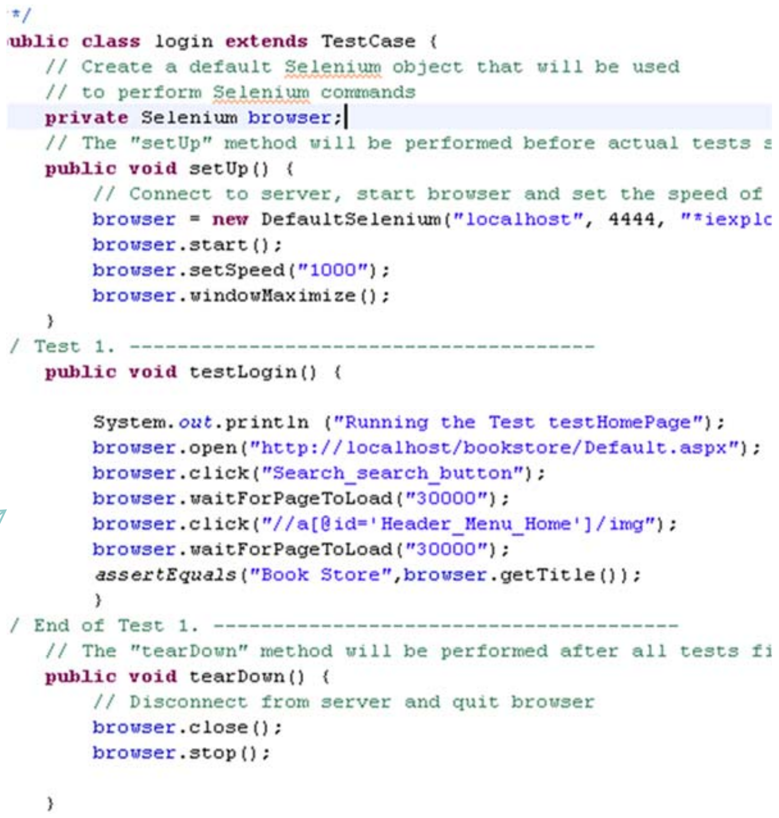

open /bookstore/Books.aspx?category_id=2&name=&
assertTextPresent Items found:
assertTitle Book Store
assertValue
assertText Total_Repeater_ctl01_item_id_Column_Title Items found:
assertTable Total_holder.2.0 Items found:
assertElementPresent Total_Repeater_ctl01_item_id_Column_Title
verifyTextPresent Items found:
verifyTitle Book Store
verifyValue
verifyText Total_Repeater_ctl01_item_id_Column_Title Items found:
verifyTable Total_holder.2.0 Items found:
verifyElementPresent Total_Repeater_ctl01_item_id_Column_Title
waitForTextPresent Items found:
waitForTitle Book Store
waitForValue
waitForText Total_Repeater_ctl01_item_id_Column_Title Items found:
waitForTable Total_holder.2.0 Items found:
waitForElementPresent Total_Repeater_ctl01_item_id_Column_Title
storeTextPresent Items found:
storeTitle Book Store
storeValue
storeText Total_Repeater_ctl01_item_id_Column_Title Items found:
storeTable Total_holder.2.0 Items found:
storeElementPresent Total_Repeater_ctl01_item_id_Column_Title
Search
Back
Forward
Reload
Stop
Bookmark This Page
Save Page As...
Send Link...
View Background Image
Select All
View Page Source
View Page Info
open /bookstore/Books.aspx?category_id=2&name=&
verifyTextPresent Items found:
verifyValue
Show All Available Commands
Test Automat...
Untitled - Not...
2 Firefox

Some Available Checkpoints

- **assertTextPresent**
  This will assert if the text is present in the page.
- **assertText**
  This will assert if a particular element is having the particular text.
- **assertTitle**
  This will assert if the page is having a proper title.
- **assertValue**
  This will assert if a Text box or check box has a particular value
- **assertElementPresent**
  This will assert if a particular UI Element is present in the page.

How Selenium Works





```
*/
ublic class login extends TestCase {
   // Create a default Selenium object that will be used
   // to perform Selenium commands
   private Selenium browser;
   // The "setUp" method will be performed before actual tests s
   public void setUp() {
       // Connect to server, start browser and set the speed of
       browser = new DefaultSelenium("localhost", 4444, "*iexplc
       browser.start();
       browser.setSpeed("1000");
       browser.windowMaximize();
   }
/ Test 1. ----------------------------------------
   public void testLogin() {

       System.out.println ("Running the Test testHomePage");
       browser.open("http://localhost/bookstore/Default.aspx");
       browser.click("Search_search_button");
       browser.waitForPageToLoad("30000");
       browser.click("//a[@id='Header_Menu_Home']/img");
       browser.waitForPageToLoad("30000");
       assertEquals("Book Store",browser.getTitle());
       }
/ End of Test 1. ----------------------------------------
   // The "tearDown" method will be performed after all tests fi
   public void tearDown() {
       // Disconnect from server and quit browser
       browser.close();
       browser.stop();

   }
```

Comparison of Main-stream Tools

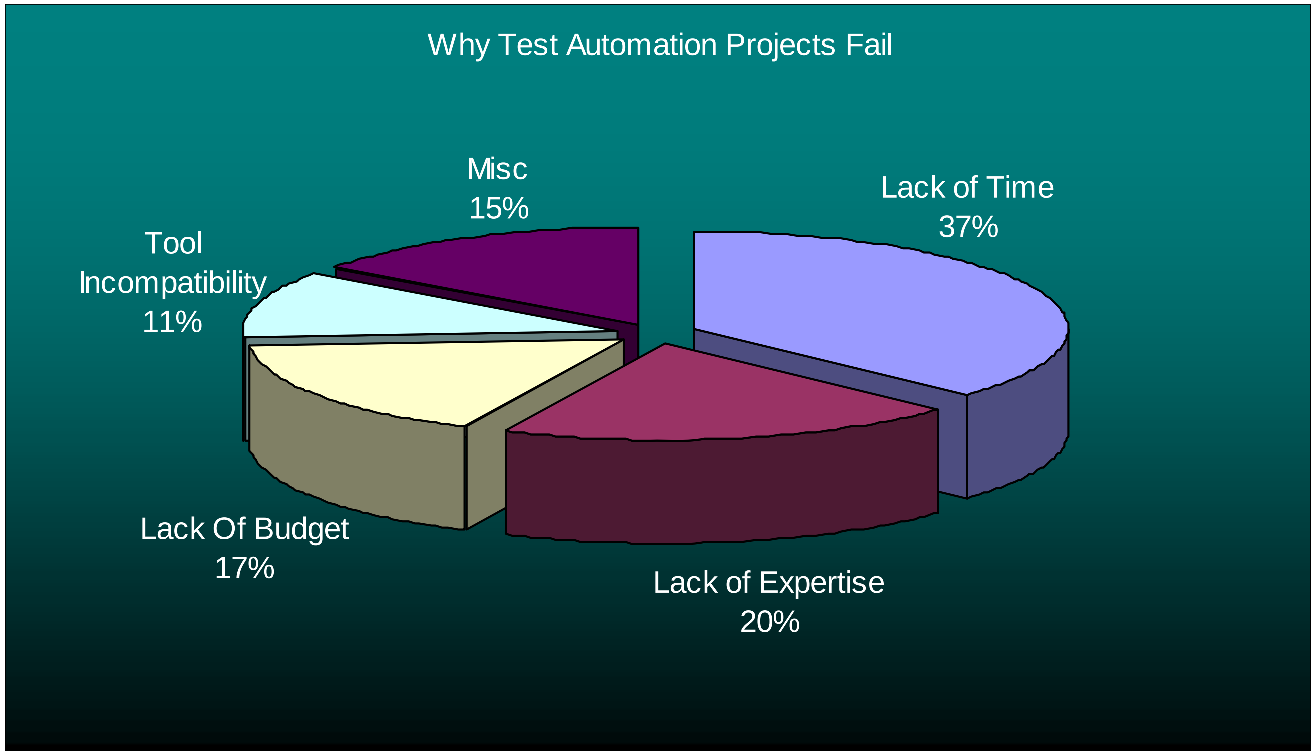
Tool	Pros	Cons
IBM/Rational Functional Tester (RFT)	•Built as Eclipse Plug-In with full IDE and Java support •Supports Web 2.0, Java or .NET applications •Full GUI Object Map repository	•Insufficient browser support •Licensed product
HP/Mercury Quick Test Pro (QTP)	•Supports Web 2.0, Java or .NET applications •Full GUI Object Map repository •Seamless integration with QualityCenter	•VisualBasic scripting is limited •No IDE (may change in new release) •Licensed Product
Selenium RC & IDE	•Good browser support •Good language support (Java, Ruby,C#) •Can be easily extended as JUnit suite •Open-source	•No GUI Object repository •Only web application support

Review: Test Automation Techniques

- Create test scripts by recording manual test execution
  - GUI object identification
  - Checkpoints to implement test oracle
- Structural Test Scripts Development
  - Use control statements and module design to improve the robustness of test scripts and enable code reuse.
- Data-driven test automation
  - Similar business process, different data. One test script to implement multiple test cases.
  - Test design is driven by tables of input data and expected output.
- Keyword-driven test automation
  - Various business processes share similar basic steps as keywords.
  - Test design is driven by a library of keywords.

Why Test Automation Project Fails

- According to a recent IDT study (www.idtus.com)



How to ensure success of TA

- Essentially automated test system is a piece of software designed to test your application → treat it as such
  - Evaluate an automated tool
    - POC (proof of concept) based on your native apps
    - Choose a tool that fits your business requirements & technical expertise
  - Plan
    - Establish goals & objectives; budget; use or hire competent resources
    - Be ready to have quality metrics to validate the ROI (return of investment)
  - Design
    - Keep in mind the user community of TA system (Usability)
    - Choose an architecture that reduces maintenance and is extensible
  - Implement
    - Practice OOD principles of reusability & modularity (employ encapsulation & abstraction)
    - Introduce version control of your code
    - Establish code review process
  - Test & Maintain
    - Unit test your code
    - Constantly re-evaluate reasons for maintenance and improve TA architecture to minimize maintenance

Thank you!