Software Test & Performance CONFERENCE

October 19 - 23, 2009

# First Steps to Test Automation

Robert Walsh - President - EnvisionWare, Inc.
rwalsh@envisionware.com



Copyright©2008-2009 Robert Walsh - All Rights Reserved



## Introductions

- Robert Walsh
  - President - EnvisionWare, Inc.
  - Developer
  - Agile Proponent
    - Especially Test-Driven Development (TDD) and Incremental Development

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE



# Introductions

- Name
- Title or Role
- City and State or Country
- What experiences or exposure do you have to test automation?
- What do you hope to learn in this class?

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE



# Agenda

- Day I - The Business Case for Test Automation
  - Section I
    - Overview
    - Pros and Cons of Automated Testing
    - Obstacles and Barriers
  - Section II
    - Types of Testing
    - Deciding What to Automate
    - Deciding How to Automate

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Agenda

- Day II - Tools to Get Started With Test Automation
  - Section III
    - Introducing AutoIt
    - Introducing Watir
  - Section IV
    - Introducing FitNesse
    - Conclusion

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

# First Steps to Test Automation

## Section I



Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Overview

- Goals and Objectives
  - To identify the pros and cons of automated testing
  - To identify and describe types of testing, highlighting those where automation is appropriate
  - To identify and describe techniques for automated testing
  - To introduce some inexpensive (or free) testing tools and demonstrate how they are used

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Overview

- Ground Rules
  - Participate
    - Ask questions
    - Share ideas and experiences
    - Raise doubts and concerns

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Pros and Cons of Automated Testing

Software Test & Performance CONFERENCE

9

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Pros and Cons of Automated Testing

Software Test & Performance CONFERENCE

10

- Small group discussion
  - Groups of 2 or 3
  - If you have experience with automated testing
    - What really "worked"?
    - What didn't go so well?
  - If you do not have experience with automated testing
    - What do you most hope to gain?
    - What has you scared or intimidated?

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Pros and Cons of Automated Testing



- Test results are easily made visible
  - Output or summary of each run may be published
    - via email
    - on an intranet
  - May be necessary to educate others on how to read and properly interpret the results
  - Transparency may make some uncomfortable

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Pros and Cons of Automated Testing



- Effort invested in automation is recovered incrementally with each run of the automated test
  - Must be valuable to run the same test many times
  - Either
    - Implies that certain aspects of the application remain static for long periods of time
  - Or
    - Requires a flexible automation framework that mitigates variance in input and output

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Pros and Cons of Automated Testing

Software Test & Performance CONFERENCE



- More tests can be run per period of time, presumably yielding better test coverage
  - However, "more tests" may mean excessive coverage in some areas and no coverage in others

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Pros and Cons of Automated Testing

Software Test & Performance CONFERENCE



- Tests are run the same way every time
  - Reduces false negatives from tester error
  - Improves efficiency when doing regression testing
  - May be difficult to simulate random or unpredictable user behavior

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Pros and Cons of Automated Testing



- Automated tests can serve as a form of documentation
  - Tests become "executable requirements" that help to supplement traditional requirements and use cases
  - Tests as documentation have less risk of getting out of sync with the product being tested
  - May require a paradigm shift on the part of management and Business Analysts
  - Executable requirements may not be "portable"
    - May require some sort of duplication or facsimile for certain uses
      - i.e., to take into a meeting

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Pros and Cons of Automated Testing



- What else?

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

17

# Obstacles and Barriers

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

18

# Obstacles and Barriers

- Small group discussion
  - Groups of 2 or 3
  - If you have experience with automated testing
    - What were some of the initial struggles?
  - If you do not have experience with automated testing
    - What are your barriers to entry?

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Obstacles and Barriers

- Time
  - Difficult to decide to invest tester time to build automation
    - The time a tester spends building automation is time the tester is not spending actually testing
    - Initial impact may be fewer tests executed than with manual testing

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Obstacles and Barriers

- Cost
  - Testing tools and the people who know how to use them are often expensive

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Obstacles and Barriers

- Learning Curve
  - Automation often requires scripting
    - "Open" languages like Javascript, Perl, etc.
    - Proprietary tool-based languages

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Obstacles and Barriers

- Effort associated with maintenance
  - Existing tests must stay in sync with the application
  - New tests must be continuously added as new functionality is developed

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Obstacles and Barriers

- Dependence on developers
  - Good test automation often requires hooks or APIs to be built into the application

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Obstacles and Barriers

- Complexity
  - If tests are code, who is testing the tests?

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Obstacles and Barriers

- What else?

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Exercise

- Making the Business Case
  - Groups of 2-3
  - Drawing from your own experiences and your personal situations, prepare a business case to convince your organization to adopt test automation
    - Sell the benefits
    - Explain how drawbacks will be addressed
    - Attempt to quantify startup and ongoing costs and efforts in order to provide justification to the business
      - e.g. "By investing __ hours (or $$) in automation, we will be able to _____."

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

# First Steps to Test Automation

Section II



Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE



## Types of Testing

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Types of Testing

- Two ways to categorize
  - Testing activities
  - General techniques for performing automation

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Types of Testing
## Testing Activities

- Unit Tests

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE



# Types of Testing

## Testing Activities

- Unit Tests
  - Generally performed by developers as part of the development process
    - Before every check in
    - As part of the compilation
  - Tests need to run fast so they will be run frequently

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE



# Types of Testing

## Testing Activities

- Unit Tests
  - Tests written in the same language as the application
  - xUnit family
    - jUnit, nUnit, cppUnit, etc.

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

33

# Types of Testing

## Testing Activities

- Unit Tests
  - Focus is on discrete and isolated functionality
    - A single module or subsystem
    - External components often stubbed or simulated

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

34

# Types of Testing

## Testing Activities

- Unit Tests
  - Test-first (also called Test-Driven Development or TDD) is where a test is written *before* new functionality is added
    - Developer then writes just enough functionality to pass the test
    - When the last test passes, the developer writes another test before adding any more functionality
    - Tests are usually written by the same developer who writes the code
      - With Pair Programming, one developer may write the test and the other writes the code

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

35

# Types of Testing

## Testing Activities

- Unit Tests
  - Test-second is where tests are written *after* functionality has been implemented to verify that the implementation is correct
    - Tests and code may be written by the same developer or by completely separate groups

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

36

# Types of Testing

## Testing Activities

- Unit Tests
  - One of the easiest areas in which to start test automation

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Types of Testing
## Testing Activities

- Acceptance Tests

Software Test & Performance CONFERENCE

37

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Types of Testing
## Testing Activities

- Acceptance Tests
  - Tests serve to prove that key features behave as desired
  - May not cover every corner case or error condition

Software Test & Performance CONFERENCE

38

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

39

# Types of Testing

## Testing Activities

- Acceptance Tests
  - Tests are specified in the language of the Customer
    - Customer must be able to understand what is being tested
    - Customer should be able to participate in writing these tests
  - Often used in Agile environments to confirm that selected functionality was implemented as part of an iteration

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

40

# Types of Testing

## Testing Activities

- Acceptance Tests
  - May require assistance from Development
    - Some Acceptance Testing frameworks require sets of fixtures to bind high-level tests to low-level implementation

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

41

# Types of Testing

## Testing Activities

- Acceptance Tests
  - May simulate some external components
    - In order to control the testing environment
    - Because certain elements may not yet be fully implemented

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

42

# Types of Testing

## Testing Activities

- Integration Tests

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE



# Types of Testing

## Testing Activities

- Integration Tests
  - Test that the system functions as a whole

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE



# Types of Testing

## Testing Activities

- Integration Tests
  - Attempt to more closely emulate user behavior
  - May strive to validate the complete user experience from installation through each individual task or work flow

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

45

# Types of Testing

## Testing Activities

- Integration Tests
  - Access to external components is fully implemented through test servers, sample databases, etc.

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

46

# Types of Testing

## Testing Activities

- Integration Tests
  - Less dependence on Development
    - Application is more complete
    - Goal is to use the system as a user would use it

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Types of Testing
## Testing Activities

- Performance / Load Tests

Software Test & Performance CONFERENCE

47

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Types of Testing
## Testing Activities

- Performance / Load Tests
  - Seek to determine how the system behaves when stressed
  - May be used to define (rather than validate) criteria like minimum system requirements, response times, maximum concurrent users, etc.

Software Test & Performance CONFERENCE

48

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

49

# Types of Testing

## Testing Activities

- Performance / Load Tests
  - Generally good candidates for automation
    - Often difficult to simulate realistic user loads and concurrent users manually
    - May leverage APIs via scripts rather than testing directly through the UI
    - May take advantage of virtual computers rather than lots of physical hardware

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

50

# Types of Testing

## Testing Activities

- Exploratory Tests

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

51

# Types of Testing

## Testing Activities

- Exploratory Tests
  - Designed to *explore* how the system behaves when a user does strange things
    - Really unusual user input
    - Unique combinations of actions
    - Stuff "no user would ever do"

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

52

# Types of Testing

## Testing Activities

- Exploratory Tests
  - Not generally good candidates for automation
  - However, effective automation facilitates effective exploratory testing
    - Automated tests provide regression, "happy path" coverage, and verification for "expected" error conditions
    - Testers are more free to focus on "what if" style manual testing

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

53

# Types of Testing

## Testing Activities

- Exploratory Tests
  - Provide excellent opportunities for talented testers to demonstrate value
    - May uncover cases no one ever thought about
    - Failures found through exploratory testing may be converted into new tests that can be automated
      - Automated regression gets better
      - Tester is again free to move on to more interesting or unusual situations

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

54

# Types of Testing

## Testing Activities

- Usability Tests

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

55

# Types of Testing

## Testing Activities

- Usability Tests
  - Seek to determine whether
    - The system is useful
    - The system is intuitive
    - The system can be used effectively and efficiently by the target audience

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

56

# Types of Testing

## Testing Activities

- Usability Tests
  - May measure
    - Whether pertinent information is displayed effectively
    - How many steps, clicks, or keystrokes are required to perform individual tasks
    - How much training certain types of users are likely to require in order to use the system
    - The variance between novice and expert users

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

57

# Types of Testing

## Testing Activities

- Usability Tests
  - Tend to be somewhat subjective
    - May be based on
      - Academic or corporate studies
      - User feedback through interviews or surveys

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

58

# Types of Testing

## Testing Activities

- Usability Tests
  - Generally not good candidates for automation
    - Unless the criteria used to determine usability can be quantified and measured with available technology

Copyright©2008-2009 Robert Walsh - All Rights Reserved

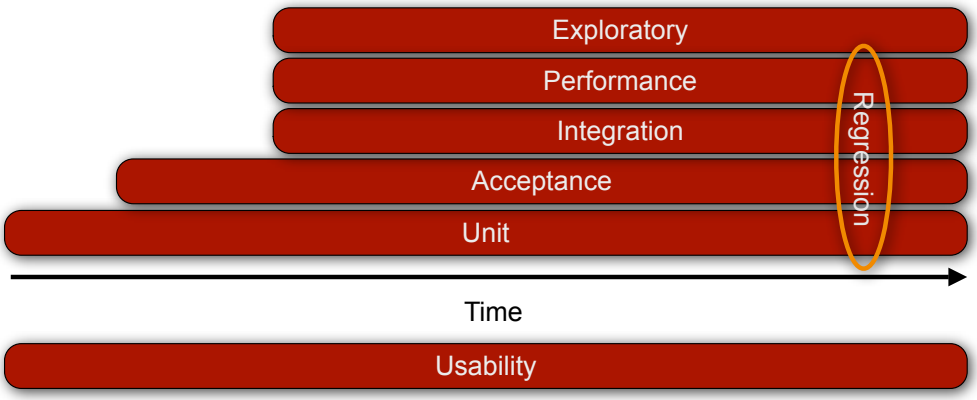
# Types of Testing
## Testing Activities

- Usability Tests
  - May not require a completed application to perform
    - Might use wireframe UIs, storyboards, or other mock-ups
  - May be done at any stage of the project

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Types of Testing
## Testing Activities




Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

61

# Types of Testing

## Testing Activities

- What other types of testing do you use?

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

62

# Types of Testing

## General Techniques for Performing Automation

- Record and Playback

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Types of Testing

## General Techniques for Performing Automation

- Record and Playback
  - A person uses the application by following a prepared script
  - All of the user's actions are recorded and may be played back later to redo the test

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Types of Testing

## General Techniques for Performing Automation

- Record and Playback
  - The timing of the user's interactions may be preserved or it may be compressed
    - Either might be valuable under certain circumstances

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

65

# Types of Testing

## General Techniques for Performing Automation

- Record and Playback
  - Probably the least expensive way to automate
    - Tests are the by-product of manual testing rather than requiring lots of dedicated preparation
    - May not require any special skills or knowledge
    - Will require tool or framework
    - May provide limited Return on Investment
  - Requires no assistance from Development
    - All testing is done through the same UI that a user will use

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

66

# Types of Testing

## General Techniques for Performing Automation

- Record and Playback
  - Tend to be fragile
    - Often depend on precise placement of UI elements
      - Minor changes can break existing tests
      - May have to re-record in order to capture new desired behavior
    - May be affected by screen resolution
      - Tests recorded at one resolution may not execute properly at another
  - May work for applications with very stable interfaces where the same action may be performed any number of times with the same expected outcome

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE



# Types of Testing

## General Techniques for Performing Automation

- Scripted Control of User Interface

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE



# Types of Testing

## General Techniques for Performing Automation

- Scripted Control of User Interface
  - Like Record and Playback in that testing is done through the user interface
    - Some Record and Playback tools generate scripts that can be edited and generalized

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

69

# Types of Testing

## General Techniques for Performing Automation

- Scripted Control of User Interface
  - Tend to be more flexible than Record and Playback
    - Selection of UI elements generally done by accessing named controls rather than based on screen position
      - More resilient to change
      - Requires some assistance from Development
    - Timing is generally controlled via sleep or wait calls in the scripts
    - Provides the ability to adapt to changes in the application by maintaining the scripts

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

70

# Types of Testing

## General Techniques for Performing Automation

- Scripted Control of User Interface
  - May be more costly that simple Record and Playback
    - Testers may need scripting skills and the ability to "think like a programmer"
    - Scripts require maintenance
  - Will require tool or framework

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

71

# Types of Testing

## General Techniques for Performing Automation

- Scripting through Public API

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

72

# Types of Testing

## General Techniques for Performing Automation

- Scripting through Public API
  - Scripts exercise the application by making the same calls the "real" client applications will make

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Types of Testing

## General Techniques for Performing Automation

- Scripting through Public API
  - Useful for web services, CGI scripts, etc.
  - Could be used to test static web pages
    - Execute HTTP GET and analyze resulting HTML
  - Tends to work well in situations where the application is completely independent of the actual user interface
  - May be a good choice for Performance Testing

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Types of Testing

## General Techniques for Performing Automation

- Scripting through Public API
  - Requires assistance from Development
    - Need thorough and well-defined API

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Types of Testing

## General Techniques for Performing Automation

- Scripting through Public API
  - Possible to build value quickly, especially if the API is stable
    - Existing scripts often require little maintenance
  - May be possible without a special tool
    - Most APIs called through libraries or over sockets

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Types of Testing

## General Techniques for Performing Automation

- Scripting through Private API or Special Fixtures

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

77

# Types of Testing

## General Techniques for Performing Automation

- Scripting through Private API or Special Fixtures
  - Application exercised by leveraging special access not intended for public use

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

78

# Types of Testing

## General Techniques for Performing Automation

- Scripting through Private API or Special Fixtures
  - Requires assistance from Development
    - Unlike with a public API, there may be no tangible business need for this access outside of testing
      - May have to "sell" the idea to management
      - May have to formalize the idea that the testers are the customer for the API

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

79

# Types of Testing

## General Techniques for Performing Automation

- Scripting through Private API or Special Fixtures
  - Sometimes used to test "just below the UI"
    - Rather than driving the GUI directly, test scripts inject the input that would have come from the user
    - Behavior is verified by looking at what the application would have done to the GUI and by confirming that all required side-effects took place
    - Requires strict separation of UI and business logic
      - Model-View-Presenter pattern

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

80

# Types of Testing

## General Techniques for Performing Automation

- What other techniques have you used?

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

81

# Deciding What to Automate

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

82

# Deciding What to Automate

- How do we decide where to start?

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Deciding What to Automate

- The Great Debate
  - Some feel that we should attempt to automate everything
  - Others feel that automation will never be as effective as manual testing with a skilled tester
- What do you think?

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Deciding What to Automate

- Seriously consider automated unit testing
  - Potential for relatively high value with relatively low cost
  - Few drawbacks or negative consequences
  - Unfortunately, not usually under the control of the QA group

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Deciding What to Automate

- Do you have support and buy-in from the Business?
  - Think about Automated Acceptance Testing as a means for determining
    - Incremental earned value
    - Suitability for release

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Deciding What to Automate

- Are data throughput and system response times important?
  - Think about automating Performance Tests in order to
    - Measure observed performance against requirements
    - Define system requirements necessary to support desired loads

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Deciding What to Automate

- Decisions about how to apply automation efforts should balance
  - Cost
    - How much are we willing to invest?
  - Value
    - How much will we save through efficiency or greater test coverage?
  - Risk
    - What price do we pay when we release with defects?

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Deciding What to Automate

- Like most business decisions, we want maximum value for minimum cost within acceptable risk

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

89

# Deciding What to Automate

- What other factors influence decisions about what to automate?

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

90

# Deciding How to Automate

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE



# Deciding How to Automate

- How do we decide what automation techniques to employ?

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE



# Deciding How to Automate

- Take an inventory
  - What skills do I have on my team?
  - Can I add skills through hiring or training?
  - What tools (or budget for tools) do I have at my disposal?
  - What kind of an application am I testing?
  - What support do I have from Development?

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Deciding How to Automate

- What skills do I have on my team?
  - Almost all automation except for Record and Playback requires some form of scripting
  - Some testing tools require specialized knowledge and experience
    - Sometimes, this knowledge may transfer to other tools

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Deciding How to Automate

- Can I add skills through hiring or training?
  - Hiring is often viable only as a solution for general, long-term needs
    - Recruiting can be unpredictable
  - Effectiveness of training often depends on
    - Relevance
    - Timeliness
    - Reinforcement
  - Both can be good ways to raise the baseline skill level within the team

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Deciding How to Automate

- What tools (or budget for tools) do I have at my disposal?
  - Commercial tools can be very expensive
    - Acquisition costs
    - Training
  - If you have them, try to find ways to use them
    - Learn what the tools are capable of doing
    - Be careful not to force your testing to fit the tool

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Deciding How to Automate

- What kind of an application am I testing?
  - Traditional thick client
  - Browser-based
  - Web services
  - Developer libraries
  - Embedded
  - Mission critical
  - Each type of application may be differently suited to a particular type of automation

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Deciding How to Automate

- What support do I have from Development?
  - Will they build in and maintain hooks and fixtures?
  - Will they collaborate on and abide by naming conventions for UI controls?
  - The degree to which Development will cooperate may dictate what types of automation you are able to consider

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Deciding How to Automate

- Automation may not be a "one size fits all" proposition
  - Try to employ the most effective technique for the situation based on particular needs, assets, and objectives
  - Look for ways to use automation to get more value out of manual testing efforts
  - Overlapping strategies may provide cumulative benefits and help to minimize "gaps"
    - However, employing more than one type of automation will likely increase costs

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Deciding How to Automate

- What other factors should be considered when deciding how to automate?

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

## First Steps to Test Automation

Section III



Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

101

# Introducing AutoIt

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

102

# Introducing AutoIt

- http://www.autoitscript.com/autoit3/
- From the AutoIt website:

> AutoIt v3 is a freeware BASIC-like scripting language designed for automating the Windows GUI and general scripting. It uses a combination of simulated keystrokes, mouse movement and window/control manipulation in order to automate tasks in a way not possible or reliable with other languages (e.g. VBScript and SendKeys). AutoIt is also very small, self-contained and will run on all versions of Windows out-of-the-box with no annoying "runtimes" required!

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

103

# Introducing AutoIt

- Classified as "Scripted Control of UI"
  - Well-suited for testing traditional thick client applications
  - Can drive Internet Explorer
  - Available only for Windows
  - Controls are accessed via names, ids, or screen position
  - Timing may be controlled using a sleep function
  - Scripts may be divided into modules and functions
    - Promotes re-use by allowing for the creation of shared low-level and utility functions
  - Not really designed as a testing tool

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

104

# Introducing AutoIt

- Test Development Environment
  - Context-sensitive IDE (Customized version of SciTE)
  - Utility for exploring UI in application under test
  - Thorough help with examples
  - Tests may be run from the IDE or as self-contained compiled executables
    - Output may be displayed or captured
      - Through GUI elements like MessageBoxes and Dialogs
      - By writing to the console (with redirection)
      - By writing to a log file

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Introducing AutoIt

- "Gotchas"
  - May not work well with custom GUI frameworks
    - Qt (http://www.trolltech.com)
    - Java
    - Some controls inaccessible from AutoIt
    - May have to revert to screen position

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Introducing AutoIt

- Sample script
  - Identifying controls

```
;by control id
$MemoryAdd = 116

;by control name
$Backspace = "Button22"

;by class and instance
$Zero = "[CLASS:Button; INSTANCE:8]"
```

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Introducing AutoIt

- Sample script
  - Running the tests

```
;Define which tests should be run
Test_Add_71_and_2()
Test_ClearLastNumberEntered()

$message = "Tests completed successfully"
MsgBox(4096, "Success", $message, 3)
ConsoleWrite($message & @CRLF)
```

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Introducing AutoIt

- Sample script
  - Running the tests

```
;Test functions
Func Test_Add_71_and_2()
  Start("Calc.exe", $ApplicationName)
  PressAndCheck($Seven, "7. ")
  PressAndCheck($One, "71. ")
  PressAndCheck($Plus, "71. ")
  PressAndCheck($Two, "2. ")
  PressAndCheck($Equals, "73. ")
  WinClose($ApplicationName)
EndFunc
```

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Introducing AutoIt

- Sample script
  - User-defined functions

```
Func Start($path, $windowText)
     Run($path)
     WinWaitActive($windowText)
EndFunc

Func PressAndCheck($control, $expectedDisplay)
     Press($control)
     Confirm($Display, $expectedDisplay)
EndFunc

Func Press($control)
     ControlClick($ApplicationName, "", $control)
     Sleep(1)
EndFunc

Func Confirm($control, $expectedValue)
     $actualValue = ControlGetText($ApplicationName, "", $control)
     If $actualValue <> $expectedValue Then
          Fail($expectedValue, $actualValue)
     EndIf
EndFunc
```

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Introducing AutoIt

- Demonstration

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

111

# Introducing AutoIt

- Installing AutoIt

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

112

# Introducing AutoIt

- Lab

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

113

# Introducing Watir

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

114

# Introducing Watir

- http://wtr.rubyforge.org/ (or http://www.watir.com)
- From the Watir website:

  Watir is an open-source library for automating web browsers. It allows you to write tests that are easy to read and maintain. It is simple and flexible.

  Watir drives browsers the same way people do. It clicks links, fills in forms, presses buttons. Watir also checks results, such as whether expected text appears on the page.

  Watir is a family of Ruby libraries. They support Internet Explorer on Windows, Firefox on Windows, Mac and Linux, Safari on Mac, Chrome on Windows and Flash testing with Firefox.

  Like other programming languages, Ruby gives you the power to connect to databases, read data files and spreadsheets, export XML, and structure your code as reusable libraries. Unlike other programing languages, Ruby is concise and often a joy to read.

  Watir stands for "Web Application Testing in Ruby". It is pronounced *water.*

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Introducing Watir

- Classified as "Scripted Control of UI"
  - Well-suited for testing browser-based applications
    - Provides scripted access to Internet Explorer and Firefox
  - Available for Windows, Macintosh, and Linux
  - Controls are accessed via names, ids, or relative position within the HTML
    - Full access to the DOM

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Introducing Watir

- Scripting is done in Ruby
  - Provides tremendous scripting power
  - May have a learning curve
  - A .net port is available
    - Watin: http://watin.sourceforge.net/
  - A Java port is available
    - Watij: http://sourceforge.net/project/showfiles.php?group_id=165206&package_id=187350&release_id=569095
    - Pronounced "wattage"

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Introducing Watir

- Test Development Environment
  - Basic text editor (SciTE)
  - Basic tutorial available on-line
  - Ruby help tools
    - irb
      - *object*.inspect
      - *object*.methods
    - rdoc, ri, fxri
    - On-line Ruby resources
  - Access to built-in rspec, runit
  - Tests run through Ruby interpreter
    - Output written to console or log file

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Introducing Watir

- "Gotchas"
  - Installation instructions may be incomplete
    - Had to manually perform extra steps
      - gem install win32-process
      - gem install win32-api
      - gem update windows-pr
  - Note: These issues appear to be resolved in the current release

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Introducing Watir

- Sample script

```
require 'watir'
url = "http://www.google.com"
ie = Watir::IE.new
ie.goto url
ie.text_field(:name, "q").set("Automated testing")

submitTime = Time.now
ie.button(:name, "btnG").click
responseTime = Time.now - submitTime
puts "Response time: " + responseTime.to_s + " seconds"
pageText = ie.html
matches = pageText.scan(/<[B|b]>([\d|\.]*)<\/[B|b]> seconds/)

ie.close
```

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Introducing Watir

- Demonstration

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

121

# Introducing Watir

- Installing Watir

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

122

# Introducing Watir

- Lab

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

# First Steps to Test Automation

Section IV



Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE



## Introducing Fitnesse

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Introducing FitNesse

- http://www.fitnesse.org/
- Excerpts from the FitNesse wiki:

***FitNesse* is a software development collaboration tool**

FitNesse enables customers, testers, and programmers to **learn what their software should do**, and to automatically compare that to **what it actually does do**. It compares customers' expectations to actual results.

**FitNesse is a software testing tool.**

From another perspective, FitNesse is a lightweight, open-source framework that makes it easy for software teams to:

- Collaboratively define AcceptanceTests -- web pages containing simple tables of inputs and expected outputs.
- Run those tests and see the results.

**FitNesse is a wiki.**

- You can easily create and edit pages.

**FitNesse is a web server.**

- It requires **no configuration or setup**.
- Just run it and then direct your browser to the machine where it is running.

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Introducing FitNesse

- Classified as "Scripting through Private API or Special Fixtures"
  - Customer (with Business Analysts and QA) write acceptance tests in FitNesse wiki language
    - Some HTML
    - Some wiki shortcuts
    - Lots of plain text
      - Many of the tests can be structured to read like a natural language requirement
  - Developers create fixtures to bind tests to objects in the application under test
  - Testing is done "just below the surface" rather than directly through the UI

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Introducing FitNesse

- Excellent choice for testing business rules and logic
- Encourages modularity in program design
  - Application needs to be structured to facilitate testing through the fixtures
- Promotes the idea of an "executable specification"
- Fixtures written in "real" programming language
  - FIT servers available for Java, Ruby, C#, C++, and more
  - No practical limitations on what can be tested
  - Requires constant collaboration with Development

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Introducing FitNesse

- Sample script

1

*Set Up: .ProductsList.BarCodePlusSuite.SetUp (edit)*

| script | | | |
|---|---|---|---|
| setup_user | 9995 | with_balance | 1025 |
| create_simulator | 4589 | | |
| set_preference | Price 4 | to | 30 |

2

*Included page: StartBcp (edit)*

| script |
|---|
| start_simulator |
| install_bcp_host |
| start_bcp_host |
| wait_for_handshake 5 |

3

*Included page: StartBcp (edit)*

| script | | |
|---|---|---|
| check_time | | |
| check | price_line_pulse_lengths | [50, 0, 0, 0] |
| check | price_line_blind_lengths | [650, 0, 0, 0] |
| check | page_costs | [15, 15, 15, 30] |
| check | location | mcr_simulator_4589 |
| check | offline_transactions | [] |
| check | cursor_position | [2, 12] |
| check | echo_position | [4, 1] |
| check | key_entry_mode | 1 |
| check | key_entry_option | 3 |
| check | key_timeout | 5 |

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Introducing FitNesse

- Sample script

5

Included page: StartLogin (edit)

| script | | | | |
|---|---|---|---|---|
| check | display_balance_row | 3 | | |
| check | copy_session_inactivity_timeout | 60 | | |
| check | copy_session_active | true | | |
| check | currency_symbol | $ | | |
| check | currency_decimal_places | 2 | | |
| check | account_balance | 1025 | | |
| check | display_line_1 | Copier Enabled_____ | | |
| check | display_line_2 | Balance:_____ | | |
| check | display_line_3 | _____$ 10.25____ | | |
| check | display_line_4 | <CLEAR> to end_____ | | |
| make | 3 | copies_on_line | 2 | |
| press | <clear> | | | |
| wait | 5 | | | |
| check | copy_session_active | false | | |
| check | display_line_1 | Scan card or_____ | | |
| check | display_line_2 | type number_____ | | |
| check | display_line_3 | _____ | | |
| check | display_line_4 | _____ | | |
| check | user | 9995 | balance | 980 |

4

Included page: StartLogin (edit)

| script | | |
|---|---|---|
| check | display_line_1 | Scan card or_____ |
| check | display_line_2 | type number_____ |
| check | display_line_3 | _____ |
| check | display_line_4 | _____ |
| press | 9995 | |
| wait_for_update | 2 | |

6

| Query:print transactions for | 9995 | | | |
|---|---|---|---|---|
| release station | cost | pages | paid from | client |
| ET380-mcr_simulator_4589 | 45 | 3 | 1 | ET380@127.0.0.1 |

Tear Down: .ProductsList.BarCodePlusSuite.TearDown (edit)

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Introducing FitNesse

- Sample script

Included page: StartBcp (edit)

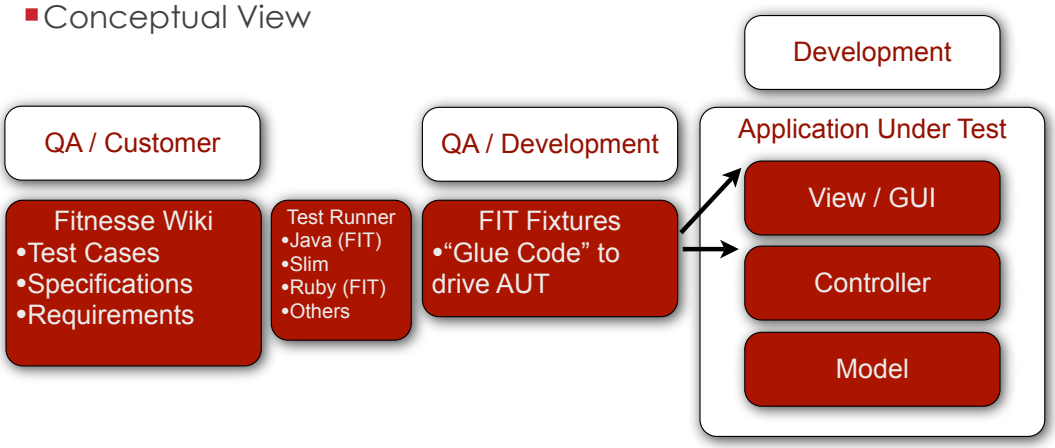
| script | | |
|---|---|---|
| check_time | | |
| check | price_line_pulse_lengths | [50, 0, 0, 0] |
| check | price_line_blind_lengths | [650, 0, 0, 0] |
| check | page_costs | [15, 15, 15, 30] |
| check | location | mcr_simulator_4589 |
| check | offline_transactions | [] |
| check | cursor_position | [2, 12] |
| check | echo_position | [4, 1] |
| check | key_entry_mode | 1 |
| check | key_entry_option | 3 |
| check | key_timeout | 5 |

```
!include -c StartBcp
|script|
|check_time|
|check|price_line_pulse_lengths|[50, 0, 0, 0]|
|check|price_line_blind_lengths|[650, 0, 0, 0]|
|check|page_costs|[15, 15, 15, 30]|
|check|location|mcr_simulator_4589|
|check|offline_transactions|[]|
|check|cursor_position|[2, 12]|
|check|echo_position|[4, 1]|
|check|key_entry_mode|1|
|check|key_entry_option|3|
|check|key_timeout|5|
```

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Introducing FitNesse

- Conceptual View




Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Introducing FitNesse

- Demonstration

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

133

# Introducing FitNesse

- Installing FitNesse

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

134

# Introducing FitNesse

- Lab

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Introducing FitNesse

- Bonus Material
  - AutoIt has a command-line interface and writing a FitNesse fixture to drive it is trivial
    - Example: auto_it.rb
  - Watir can be driven through FitNesse with a tool called Faucets (http://faucets.rubyforge.org)
  - Watin can be driven through FitNesse with a Watin plug-in for FitNesse

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Conclusion

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Conclusion

- Benefits of Automated Testing

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Conclusion

- Benefits of Automated Testing
  - Test results easily made visible
  - Incremental cost recovery
  - More tests may result in better coverage
  - Consistency in the way tests are run
  - Executable specifications

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Conclusion

- Obstacles and Barriers

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Conclusion

- Obstacles and Barriers
  - Time
  - Cost
  - Learning curve
  - Test maintenance
  - Dependence on developers
  - Tests may become complex

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Conclusion

- Testing techniques that are candidates for automation

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Conclusion

- Testing techniques that are candidates for automation
  - Unit
  - Acceptance
  - Integration
  - Performance
  - Collectively: Regression

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Conclusion

- Testing techniques that may be best done manually

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Conclusion

- Testing techniques that may be best done manually
  - Exploratory
  - Usability
- Effective automation facilitates better manual testing

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE



# Conclusion

- Techniques for Automated Testing

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE



# Conclusion

- Techniques for Automated Testing
  - Record and Playback
  - Scripted control of the User Interface
  - Scripting through a Public API
  - Scripting through a Private API or Special Fixtures

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Conclusion

- Deciding What to Automate

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Conclusion

- Deciding What to Automate
  - Look at the support available from other groups
    - Unit Tests - need support from Development
    - Acceptance Tests - need support from the Business
  - Look at what is important
    - Performance Tests - value depends on importance of requirements like response times and maximum load
  - Look at what is repetitive
    - Regression Tests
  - Need to balance cost, value, and risk

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Conclusion

- Deciding How to Automate

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Conclusion

- Deciding How to Automate
  - Take an inventory
    - What skills do I have on my team?
    - Can I add skills through hiring or training?
    - What tools (or budget for tools) do I have at my disposal?
    - What kind of an application am I testing?
    - What support do I have from Development?
  - Mix and match techniques to maximize value and minimize gaps

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Conclusion

- AutoIt

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Conclusion

- AutoIt
  - Scripted control of User Interface
    - Proprietary but relatively simple scripting language
  - Available only for Windows
  - Good choice when you have limited support from Development

Copyright©2008-2009 Robert Walsh - All Rights Reserved

# Conclusion

- Watir

Copyright©2008-2009 Robert Walsh - All Rights Reserved



# Conclusion

- Watir
  - Scripted control of User Interface (Web Browser)
    - Scripting done with Ruby
    - Ports available for .net and Java
  - Platform independent
  - Powerful and flexible
    - May require more scripting skills than AutoIt

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE



# Conclusion

- FitNesse

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE



# Conclusion

- FitNesse
  - Scripting through Private API and Special Fixtures
    - Customer writes test in HTML/Wiki language
    - Developers write fixtures to bind tests to application under test
  - Platform independent
  - Powerful and flexible
    - Allows Business to participate in test specification
    - Requires support from Development

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

157

# Conclusion

- Questions?

Copyright©2008-2009 Robert Walsh - All Rights Reserved

Software Test & Performance CONFERENCE

158

# Thank you!

**For more information:**

Robert Walsh
President
EnvisionWare, Inc.
678-584-5911
rwalsh@envisionware.com
http://www.envisionware.com

Copyright©2008-2009 Robert Walsh - All Rights Reserved