How to Implement Full-Spectrum ‘Automation’

ELLIOTT LEVENSON

What Exactly is it You are Trying to Do?

Think Big:

Continuous Integration: Tests are automatically selected and executed as part of the Release Process (Automate the Automation run from a Script or run as a job)

- System UI Tests
- Mobile Tests (from a Hub)
- Other Functional UI Tests
- White-Box Service Tests, Data Quality Tests, Integration Tests
- Unit Tests (If desired)

Run Nightly: Or as often as desired

Ad Hoc: Extend the reach of the Tester

No Really, What are you Doing?

Think Small:

Can I get one little UI Test running?

Do it like anything else, use a good software engineering process.

Get a workflow

Design a Use Case with a path through the system based on the workflow

Write a Hello World in the chosen language API or tool

Write a simple test

Run the Test

Create a detailed How To for every step on how you did it

What are you going to do now?

Think Middle:

Analyze the environment

Design an environment

Do I want in-house agents and controllers?

Do I want ‘The Cloud’ ooh ‘The Cloud’ it’s so exciting, it’s delicious!

VMs or real machines/devices?

Do I need test executed from overseas locations?

What is your security/network architecture?

Open Stack? Proprietary? NOTHING WORKS OUT OF THE BOX!

Document every single environment setup and troubleshooting you did

Tool Selection

Really?

If you are thinking Tool Selection, you haven't done your homework and you are avoiding the required technical effort OUT OF FEAR OF THE UNKNOWN! OK, Let's talk UI.

Microsoft has an integrated Suite for Configuration Management Test Case Management – You can run Data-Driven Tests by getting the parameters from your test case (TFS/VSO has a little SQL Server Database hidden underneath), Test Lab (Environment), Test Execution and Development (Coded UI – by the way you can tuck your Selenium into the Coded UI [TestMethod] and run your tests automatically from Test Manager.

Open Stack: Selenium, TestNG, JUnit, Java run on Windows or Linux, use Jenkins, Eclipse, JBoss, Oracle XM, Data Driving from Excel, CSV, text File, Database, whatever, write your own test harness to grab the parameters.

UI Testing - ROI

ALL UIs WILL REQUIRE SOME MANUAL TESTS BY DEFINTION, i.e. A USER!

But a tester can’t do it all. We can save time if our UI Web Tests:

- Cross-Browser: 5+ browsers plus versions
- Platform: Desktop, Laptop, Tablet, Phone, Set Top Box, Game Console, Watch… 7+
- Localization: 12+ languages
- Operating Systems: Linux, Unix, OSX, iOS, Android, Windows plus version 5+
- Environments: We have 4
- Data-Driven: Zillions of tests, say 10
- Accessibility/Compatibility 2+

One System Test: 5x7x12x5x4x50x2 = 840,000 Tests in One! Not Human testable!

Load Testing

Hey! Let's spend a billion dollars on that cool tool! Hey! Let's not!

Microsoft Visual Studio Ultimate: Unlimited Virtual Users

- The Grinder, JMeter use a little elbow grease
- Start with one little working test (STORYBOARD GET A USE CASE! WE ARE ADULTS!)
- You can record workflow automatically or manually piece the sequence together
- Mix and match tests and browser types
- Set ramp up, number of users, length of time
- Do your bracketing: >1, Average Hourly Peak, Total Peak, Highest Ever Peak, Max Registered users, Stress brackets
- Test on Agents and Controllers or (wait for it…) 'The Cloud' ooh aah, it's so tantalizing!
- Use industry standards for the type of test (Click….wait…Click…wait)

Service Testing

Write a little pure code service test

Use some Service Testing tool like SOAPUI or ReadyAPI….

Figure out the Command line!

Then you can script all the little command lines!

Mix and match little technologies!

Run it from your release system, with lots and lots of little tests

Maybe write a test panel…

Mobile Testing

Ooh let’s a get a tool!

Let’s test in ‘The Cloud’ ooh…scintillating!

Let’s use good software engineering instead.

Windows Tablet – just throw on the Agent Software, Done if you use Microsoft

Android – Can run a test out of Test Manager by starting Appium

iOS – JavaScript based testing, Appium, Selenium, WebDriver, Jasmine, Karma, Mocha, Protractor, blah, blah, blah

Why do we Document Everything?

So we can train it! It can be done! Any tester can do it! You can train it in house! Get every tester involved. Hint: (THEY DON'T HAVE TO KNOW MASSIVE AMOUNTS OF CODING) The tester should know just enough to design an automated use case and maybe a POC/prototype. Can use Nuget packages, standard methods etc.

We train constantly:

- Test Environments, networks, ports, agents, controllers, 'The Cloud' ooh…aah…DeLovely!
- Mobile Testing, JavaScript based testing
- Load and Performance Testing, extraction, validation, dynamic objects/Web Principles Http/REST, Session etc.
- Service Testing, ABCs, etc.
- How to write a UI/Load Service Test, build it, test it deploy it run it – cradle to grave
- Object Recognition

We use any resource available and use it for a curriculum: Books, MSDN, Pluralsight, YouTube tech videos, Selenium Site, Appium Site, Product Documentation/Help, Code Documentation/Help

Start with the Product

Start with one big System Test and a Load Test for Every Product
- Use the ROI principles to get bang for the buck
- Use software engineering principles, code review, configuration management
- Template and develop standard methods

Start with your environment development at the same time

Start with one simple service test

Wire in a way to run the tests en masse or in sequence

Integrate the tests into your release process

Design a scheme for defect notification: test reporting and metrics can wait

Document and train every step, every solution, every how to

Start adding smaller tests as part of your Sprint – grow the library

Follow a defined process f/x Load Testing Benchmark, Baselines, standardize

Things I Hate

‘People’ who lack hard technical skills and don’t understand Test Automation so they duck it, avoid it, run away in fear or do ‘Tool Selection’ ooh…aah…it’s so scary!

TDD, BDD – massive overhead and waste of time, silly at best

People who don’t comment their code

People who don’t identify their objects (don’t make it so hard to test, please – solved by good code review and ‘Big Q’ QA)

Load Testing Tools from Bix Box Mega Companies (Hint: THEY’RE TOO EXPENSIVE)

Conclusion and Contact

If you want my help, just call me, but you'll have to buy me lunch, dinner or a trinket and provide free parking.

Elliott Levenson

412 303-3573

eleven@pitt.edu