Cyreath.co.uk
Empirical Pragmatic Testing

Configuring Test Environments

A rational approach to Hardware and Software for Web Testing

Discussion Document
By Mark Crowther, Empirical Pragmatic Tester

Introduction

Think about your PC, without looking, what's the version of your OS, IE or Firefox on there, how much RAM do you have and what speeds your internet connection? You probably don't know but it seems to work OK, allowing for that odd glitch right?

Now imagine this is a test system's configuration, you're sure that the bug is really a bug? Maybe that browser version you have is buggy itself or hasn't got the latest updates and patches? I can't repeat the bug on my PC, are you sure it's not your system?

Of course you can't really say, because there's no way to be reasonably assured that our versions are different or that the bugs aren't really bugs or even that there should be bugs when you see none. Add in variations of hardware and we're multiplying the possible causes of inconsistent test results. Why did you decide on that configuration anyway?

We have to consider all of these things, that's why the configuration of test hardware and software needs a rational approach.

Devise a Rational Approach

Let's leave hardware to one side for now and just think about Operating Systems and Web Browsers.

Devising the configuration for test software, with its combination of OS and browser sets, into a usable test matrix is difficult as there are many factors to consider and you want to consider these against what your customer needs.

The most important question is "What environment configuration do you expect the majority of your customers to be using your product in?"

Any strategy for devising your matrix must be based on a reasonable and rational approach and you need to ensure defined control over the test system software architecture once it's up and running. Without control your test results become meaningless as the irrational approach will provide you with no definition or control meaning uncertainty of your test results.

Baseline Your Test Software

To get some definition and control you need to start from a defined baseline. Don't be afraid either to ask your customer what they use and why. It may well be they have no clear rationale for the configurations they use. Were they just 'picked' or were they on the platform when they bought it, maybe they're not sure if they are patched or not, maybe their version is actually not representative of the version their own customers will use?

It could be many things and you need to ask them. You need to ask to get a rationale about what's going to be used to support your customers' needs.

Selecting and Controlling 3rd Party Software

We need to consider both the OS and the Browsers that our customers may be using. We have no real control over the OS or Browsers source, so let's think about how to get configurations together that will bring a consistent and agreed approach to testing.

First on your list are the most popular browsers, it's going to be IE and Firefox, then add in the others if they are relevant toy our business. We can certainly add in others but let's use IE and Firefox as worked examples.

Configuration 1 - Baseline test system architecture

Your start point is the most up to date versions of the OS and browsers as you need to define a baseline and start point. Think in terms of functionality, support and of course automatic updates, this alone means your user base is potentially getting updated every time an update is available.

Visit Microsoft and Mozilla to get the most up to date version they offer. Make sure automatic-updates is set and grab any that are made available. Security patches are a particular must especially for IE. Now turn off auto updates and make a controlled back up. Burn it to disk, copy to your CVS like system, get a Gold Master placed away or whatever you do to ensure the integrity of that browser image.

That's your baseline. It's a baseline because your customer can do the same so it's a simple to explain and repeatable approach. Getting this commonality is an often-overlooked start point! Do the same for any other configurations you want to offer testing on or your customer supports products in.

You can also supply this to the customer so you both are working in the same way. You build and test on that architecture and they accept on that architecture.

Configuration 2 – Customer required variations

Now let's assume that there are going to be other versions of browsers your customer wants to support, such as IE 6.0 or 7.0 perhaps. Do the same for these too. Get a rational and repeatable approach to building a browser images then safe source a copy for integrity's sake.

From here you can start to build up your configuration matrix, start looking at what hardware you need and perhaps connectivity too.

Reduce Variation

Now the customer needs to agree that this set of browsers, operating systems, hardware, etc. is the agreed and defined test configuration, not these and a whole bunch of 'Who knows what it is' variants.

However, it's not just to get confidence in the test results that make this approach so important. If you try to test on every possible configuration you will never create a matrix of them all and you'll probably never finish testing. Limit the variation in test configurations and limit the variation of test results.

If you struggle to get agreement on what configurations are to be tested on, just provide them the costs for testing in each configuration and let your customer choose. After all you're here to test so the more they agree to the merrier.

Your matrix should consist of valuable configurations based on business needs. It may be good to have a fully up to date and patched version, one with just the latest media player missing (if that was important to you) and maybe one that is completely unpatched, you get the idea.

However always have a core set, you can have this 'nice to have' set but don't be drawn into testing on it to declare software quality unless it's been committed to.

Should you need to alter the matrix later, you can do so with confidence about what it is you're altering them too and why.

All test configurations based on a defined, rational approach are of benefit to the test team and the business and will help guarantee the highest level of software quality.

Reference

- Check the latest Browser and OS stats and info here: http://www.w3schools.com/browsers/default.asp