

# Getting our own house in order

---

**Daryl Elfield, Managing Consultant at Starbase, believes it's no use complaining that testing is not given the importance it deserves when it's largely conducted by hard-working amateurs who happen to fit a personality profile**

*Testing has come a long way in the last few years.* As a consultant I am spending less time convincing project managers of the need for testing, and a lot more time discussing whether the testing they are doing is sufficient. This is a huge step forward and means that testing is finally being taken more seriously. However, we still have a long way to go and one of the places we can start is cleaning up our own act. It's time to ask: are we actually any good at what we do?

The problem we have as an industry is that for years now we have "got by" with minimal budgets, tight deadlines, and insufficient resources which have all led to a survival mentality – we accept what we have because we doubt it can or will get better. We accept any resources because it's better to have some testers, even if they are developers, than none at all! Unfortunately not only has this compromised the quality of our work, but it has played to the misconception held by most of the IT world – that pretty much anyone can do testing, because it's not that hard!

We have also been able to "get away" with using testers who have little or no experience. When your job consists of either pointing out flaws in other people's work or explaining all the reasons why something isn't ready to be tested or isn't ready to go live, it is rare for the quality microscope to be turned on your own work.

This is never a tenable situation long-term. Eventually project managers will or have asked: "what am I getting for my money?" and "why are you spending so long testing?" It is no longer acceptable to blame everyone else. We need to make sure we have our own house in order.

So let's start at the beginning and ask:

## **Where do testers come from?**

Most project managers and developers when faced with that question would probably say: "the seventh circle of hell" but it's a serious

question with serious consequences. And my answer, to paraphrase The Bard, is as follows: "Some are born testers, some become testers, and some have testing thrust upon them".

## **Some are born testers**

My belief is that one of the many things preventing testing from being taken seriously is the common industry view that testing is a "knack" – you either have it, or you don't. Under this theory, training or experience is irrelevant. Although it's true there are naturally gifted testers out there, they are unfortunately not representative. As professionals, we should not encourage this point of view, since it undermines the skills and experience that, in some cases, we have spent years acquiring.

---

**"We are dragging ourselves down by not insisting on training. It's time to say 'I'm mad as hell and I'm not going to take it anymore'"**

---

## **Some become testers**

How many times have you heard that a tester is just a failed developer? Other than being insulting on the face of it, it implies not only that testing is an inferior skill to development, but also that development skills are easily transferable. Unfortunately "failed developers" are not the only people beating a path into testing. We also find support people looking for a new challenge and business users who fancy their technical skills.

Is this really the best way to become a tester? How do we know whether someone

new to testing is any good? More importantly, how do we ensure they are sufficiently trained to call themselves testers? In reality, most people can become testers, but it shouldn't happen through osmosis. I'll return to this later.

## **Some have testing thrust upon them**

How many of you know of testing teams made up entirely of developers or users? "We don't have a test team, the developers test it" is a common theme as are countless variations on it. Who are these unfortunate people forced to do a job they either don't want to, or don't know how to do? And how good a job do we think they're doing? People forced to do testing will do bad testing. People forced to do testing with no training will not even know that the testing they're doing is bad!

Given this background perhaps the original question ought to be "where should testers come from?"

What makes a good tester? There are countless articles and books on this question, but they typically give an answer by defining the personality of a tester, rather than the actual skills required. Yes, a tester must be patient, have good communication skills, honesty etc. but you can have all these qualities and be a terrible tester. Why? Because at the heart of it, you don't know what you're doing!

What does "writing test scripts" mean? What does "defect logging" actually involve? Are these just things you can pick up or can you be trained? Would it make any difference if you were?

How is that such a crucial IT skill apparently requires no training? How can a company insist on "quality" products, when the very people tasked with ensuring it often have the least training? These days a developer without a computer science degree is increasingly rare, and there are any number of certifications in the various programming languages. But companies will let almost

anyone become a tester, regardless of skills or experience.

We perpetuate this situation by not insisting on training for ourselves and the members of our team. A foundation course in software testing should be a minimum requirement for obtaining a job in testing, not something to be picked up later. Would an IT manager hire a Java developer with no Java experience or training? Would a project manager let the support team "have a go" at writing C++ for his next project? It's laughable, but it's something we accept as testers every day.

Testing is not just an art. It's true that some people have a "knack" for it, just as some people have a gift for programming. But many more people can be trained in the basic skills required to be a reliable tester – requirements analysis, test scripting, test execution and defect logging. If your team do not have these skills, shouldn't this be raised as a major project risk?

#### So what makes a good tester?

We are all familiar with defending testing as a skill, and much of the need to be officially trained in testing is to convince a sceptical industry that what we know is worth knowing. But beyond that, how do we know we're actually any good at what we do unless we have a standard to measure it against? How do you

distinguish a good tester from a bad tester?

Let's go back to the beginning - where do we learn to be testers? Unfortunately our skills as testers are largely determined by the first job we take - are we surrounded by good practice, or is it a "suck it and see" operation? Hundreds of us became testers during Y2K and the dot.com boom and although we learnt to test efficiently, we never learnt the basics. I have conducted countless interviews where even fundamental questions like "what is regression testing?" or "have you ever performed document validation?" are met with blank stares.

Any actual testing skills are picked up ad-hoc, on the job. How many of us have been reduced to training our teams ourselves because the company we work for won't pick up the bill for training testers? And how good is that training if we ourselves have never been trained?

So here's what it comes down to. On a personal level, we need to get trained to be any good at our jobs. A tester with no training is just performing user acceptance testing - useful, but that's what the customer is for! If you hand me a technical specification I will not, for example, be able to conduct system testing unless I know a) what it is and b) what it consists of.

But beyond this we need to get trained so that the industry as a whole gives us the respect and credit we deserve. It is no use saying: "no-one thinks testing is important" when it is largely conducted by hard-working amateurs backed by no recognisable qualifications.

So approach your test career like you would any other IT career. Start with the basics - get the ISEB Foundation Certificate in Software Testing, even if you've been testing for years. You'll be amazed just how much you have forgotten. If you're specialising in a particular skill (eg load or automated testing), get the relevant training or certification either from the tool vendor or a qualified third party. And don't stop there: there are specialist courses in web testing, in risk analysis, LoadRunner protocols – the list is endless. This is your career, and no one will look after it as well as you will.

We are dragging ourselves down as an industry by not insisting on either training for ourselves or for our teams. Isn't it time we all said, in the immortal words of Howard Beale in the film *Network*, "I'm mad as hell and I'm not going to take it anymore?" PT



# Automated Testing. Are you maximising the benefits?

StarBase's comprehensive suite of solutions support Mercury Interactive's ever expanding range of Automated Testing and Application Management tools. Our Consulting services include:

## Load/Performance Testing

*"The Combination of StarBase testing expertise and CIMA application design knowledge assured a successful project"*  
Guy Gaskins, Chartered Institute of Management Accountants

## Functional Regression Testing

*"Within Functional Testing, StarBase have proved themselves to be much more than a service provider, they have evolved with us to be a valuable extension of our IT department".*  
Matt Cowdrey, AON

## StarBase's unique Advanced LoadRunner Training

*"Very enjoyable course with useful methodologies. We will be adopting a number of the suggested working practices"*  
Richard Green, HSBC Bank

To maximise the benefits from your Testing requirements call 0208 905 1120 or visit [www.starbase.co.uk](http://www.starbase.co.uk)



StarBase are a Mercury Interactive top tier business partner