Foreword from Mark Fewster

Ever since humans have walked our planet, we have sought to use tools to make our tasks easier and faster, and to help us extend our capabilities. Tools are an integral part of our evolution. It is no surprise then that we want to use tools to help us with testing software – it’s only natural. However, there are many more compelling reasons for us to use tools in software testing than just our natural instinct.

The business need to produce better quality software systems faster and cheaper drives our search for better ways to perform the testing, and tool use is often one of the most effective means of achieving significant improvements in each of these areas.

Testing is a challenging task that demands both thinking and effort. Thinking is necessary to analyse the system to be tested, identify and prioritise what there is to test, design tests that can reveal defects and choose when to run and re-run subsets or all of the tests. Without the thinking, testing will likely fail to find many defects, making it ineffective and yet expensive in terms of effort.

As software systems have become larger and more complex so too has the requirement for testing. This growth in the need for testing is not linear but exponential. When we add a new feature to a software system not only do we have to add tests to exercise it but we have to add tests to exercise the combinatorial explosion of interactions between the new feature and the existing features of the system. With every new feature the number of possible interactions increases. A change to one part of the software can introduce or uncover defects in other parts of the system so the scope of testing a change to the software must go beyond the scope of the change by regression testing to check that the latest version of software has not had defects introduced in the parts that worked previously. Furthermore, much of the testing may need to be repeated across multiple environments. Tests are not run once only but many times with each version of the software and throughout its useful life. Unfortunately this repetitive work can become such a large part of the testing effort that little time is left for the more creative part of testing. Consequently the quality of the testing suffers, and so too the quality of the software.

Understandably it is the repetitive effort where tools are often used first. This may be simply to ensure that sufficient retesting can be completed within an acceptable time period. It may also be to help achieve more thorough regression testing within the time period. Yet more appealing is the idea that having the test tools undertake the regression testing will allow testers to spend more of their time on the creative part of testing, thereby improving the quality of the tests and the effectiveness and efficiency of the testing as a whole (tool use can help testing find more defects).

Using test tools to support regression testing is just the first, most obvious, step on a journey that can introduce tool support into most of the testing activities.

In addition to the benefits that can be achieved by automating regression testing test tools can enable us to undertake testing that is not possible to do by manual methods alone helping us achieve richer, deeper, and more diverse testing.

It is easy to see that tool support for testing has great potential. Using tools to run tests for us can mean that they can be run in less time and at times when human testers are not normally available such as during the night and at weekends.

With all these magnificent benefits that the use of test tools can bring why isn't everyone using them?

Success with using tools is not guaranteed. Indeed many organisations have tried and failed to introduce test tools. Inappropriate and uninformed use of tools can cause damage on a grand scale. Even the simplest of tools can be misused. Think what would happen if we gave a 3-year old a hammer; a simple and effective tool, but potentially damaging and dangerous when used inappropriately. If we don't understand how to use tools wisely we can end up wasting time and resource rather than saving it. Used in the wrong way, tools may give false or misleading information.

The problem is, test tools, despite their name, do not do the testing for us. Test tools are simply tools, inanimate things, non-thinking, incapable of creative analysis. They cannot assume the responsibility for testing; they can only support the activities of people. Without human involvement they have no value.

To be successful with test tools test automation engineers, testers and test managers alike need to understand both the capabilities and the limitations of each tool used. A common cause of perceived test automation failure is unrealistic expectations. For example, a test manager who believes that the introduction of a test tool will somehow make the testing twice as effective at half the cost within the first month is probably going to be disappointed.

Test managers must also appreciate the difference between testing skills and automation skills. Building automated tests takes time and requires knowledge and skills that are different from testing knowledge and skills. Similarly the responsibilities for testing and for automation should be distinct. Test managers should ensure that individuals know when to focus on automating tests and when to focus on testing.

Introducing tool support into testing will take time and it will change the test process. This change must be managed to ensure a coordinated and consistent approach that ensures the desired benefits of tool use are achieved whilst avoiding the pitfalls.

Success with test automation is about achieving and using automated testing as part of an overall test strategy. We are talking about tool support, not a tool takeover. Skilled people will always be a vital ingredient within the testing process but as our systems have become larger and more complex, and the increased demands for high quality within less time and cost, tools are no longer optional but are also an essential ingredient in any mature test process.

How can we ensure that we use tools well and wisely? That's the purpose of this book.

Whether you're about to start test automation or are looking to improve an existing test automation effort, you're in the right place. Read on, learn from the authors – don't re-invent the wheel. Take advantage of the distilled wisdom derived from years of experience.

I wish you much success with test automation.

Mark Fewster

Grove Software Testing Ltd., 2015