# Avoiding Common Test Automation Pitfalls

By Nathan Langton, Senior Software Tester
November 5, 2013

In an age where many organizations are under pressure to accelerate their software delivery to customers, test automation is becoming a necessity. There are many reasons why it's advantageous to implement a test automation solution as part of your testing strategy: it can help shorten development cycles, it can help increase test coverage and it can increase the speed and frequency at which some tests can be executed thereby providing rapid feedback. While these benefits can be realized, successful test automation in practice is challenging to achieve.

Every software project is different and, as such, every automation solution has to be tailored to your unique situation. This means that what may have worked in some cases may not necessarily work in your circumstance. Without attempting to list every challenge you may have to overcome, this article will focus on some of the most common pitfalls that need to be navigated in order to increase the chances of success in your test automation effort.

"Every software project is different"

### *Having unrealistic expectations*

A common misconception within organizations is that test automation is the "silver bullet" for improving quality, reducing testing efforts and reducing time to market. While automated testing can have positive impacts in these areas, setting unrealistic expectations is often the most common pitfall leading to project failure. Having expectations regarding the return on investment (ROI) without performing an informed analysis of the benefits that automation can bring to your project can lead to stakeholder dissatisfaction. Additionally, placing unrealistic expectations on what can be achieved by the automation tool is a common pitfall. Prior to the start of any test automation effort, it is important to evaluate one or more toolsets to understand how they will interact with the application under test (AUT).

### *The lack of clear objectives*

Starting an automation project without clear objectives and a prepared plan is like starting out on a road trip with no idea why you are going, where you are going or how you will get there. This may seem obvious, yet it's surprising how many automation efforts begin without any clearly defined objectives that answer the questions: "Why are we undertaking this automation effort?" and "What do we hope to achieve?". A well-defined objective should be realistic, achievable and measurable. An

example of clear objectives would be to "test on different operating systems", "reduce time to market" or "increase confidence of each build". Having realistic goals that are well defined will be easier to measure and, in turn, have a higher chance of being achieved.

### *The lack of a plan*

When starting a new project, we often have the tendency to want to jump in immediately with the initial setup and coding activities while ignoring the necessary planning activities. The test automation plan and strategy documents frequently get neglected in the efforts of saving time or because there is no perceived value in writing them. The risks that result from not having a plan include the potential for scope creep, ambiguity on entry/exit criteria for various phases and budget or time overruns. It's no surprise that test automation efforts are more likely to succeed when they are well planned.

The plan should include the scope of the automation project and realistic expectations that can be achieved within the specified time frame. The plan should also help answer the following questions: "Which areas of the application under test will be automated?", "What does the framework architecture look like?", "What is the scope of testing?", "What will be delivered and when?" and "Do we need any specialized skills?". The best plan is only a guide or rough blueprint as one cannot predict all contingencies or changes that arise from the many moving parts within software development. It is for that reason that the plan should be treated as an active document that is continuously updated as the project changes.

### *The lack of a framework*

A good automation architecture or framework is what gives automation its real power and value. A framework outlines an overall structure for the various parts of the automation solution. Having a framework is not the solution in itself; it has to be designed for maintainability in order to provide value. If the maintenance is cumbersome, the updates to the framework or specific tests will not occur whenever changes are needed and this often results in a failed solution.

"The best plan is only a guide or rough blueprint"

A well-designed framework needs to be built on top of the automation tool in order for the solution to be robust, well structured, reusable and modular while employing abstraction. Separating the test data, reporting libraries, tool-specifics, utility libraries, domain-specifics, scripted tests, page models and object repositories are some of the levels of abstraction that will allow you to build reusable and maintainable code. This will ensure that the framework is maintainable for future updates which ultimately will allow any member of your team to easily identify what needs to be revised based on the changes in the AUT. Investing the time to create an automation architecture plan which outlines the design of the framework, independent of the test automation plan discussed earlier, will contribute to a successful automation effort.

### *The lack of early deliverables*

Management support for the test automation effort is imperative for its success due to the

time, effort and costs associated with automation. As such, any early feedback that demonstrates the return on investment is always needed. The development effort should be broken down into bite size deliverables that provide early ROI rather than attempting to build the full library of reusable scripts at the onset of the project. Achieving early wins with the delivery of meaningful results will help build confidence in your team and boost management support for the initiative. One successful strategy is to deliver a small subset of tests (such as smoke tests) early, prior to moving on to various modules in the full regression suite.

"One successful strategy is to deliver a small subset of tests"

This will also allow you to receive feedback on the development activities and, at the same time, demonstrate the usefulness of test automation and give your team the opportunity to provide value early into the project.

The benefits of improving quality, reducing testing efforts and reducing time to market are very much achievable with test automation. Having clear objectives, realistic expectations and an automation strategy and plan in place that outlines how you will deliver a maintainable framework will increase your chances of success. This demonstrates the importance of taking the time to provide considerable thought and planning around the automation solution and process prior to commencement of the project. In addition, learning from your past experiences and those of others will help you navigate through various challenges and steer the automation project to success.

**About The Author**

Nathan Langton is a Senior Software Tester with Professional Quality Assurance Ltd. (PQA). His past experiences include leading a team responsible for testing a financial system upgrade for a large bank, functional and automation testing activities for a large auction website and, more recently, leading an automation team responsible for building an OAuth test suite for an Identity Management system. Nathan completed both his Masters of Computer Science and his Bachelor of Computer Science degrees at the University of New Brunswick, obtaining an Honours in Software Systems. Prior to PQA, Nathan was responsible for research and development of speech technology solutions at National Research Council of Canada (NRC IIT). His research has led to three conference publications, and the opportunity to present at the prestigious MobileHCI 2008 Conference in Amsterdam.