

HOW TO
SUCCESSFULLY
AUTOMATE THE
FUNCTIONAL
TESTING PROCESS

Automated functional testing optimizes software quality and
drives business value

Contents

Executive summary	3
What is functional testing?	4
Why automate functional testing?	4
How do you successfully automate the functional testing process?	5
What are the best practices of functional test automation?	6
What should you look for in an automated functional testing solution?	7
Borland® SilkTest® – the automated functional testing solution	8
Lifecycle quality management	9
Summary	9
References	9

Executive summary

Today's software development organizations face a triple challenge. They must optimize the quality of increasingly complex software applications – and do so more quickly and cost-effectively than ever before – in order to deliver winning solutions that yield a high return-on-investment and drive competitive advantage. Borland® and its partners demonstrate commitment to tackling the most difficult challenges of software development, offering solutions that span the entire software delivery lifecycle through personalized customized solutions.

How have some software organizations responded to the challenge of complex software development? “Automation is the answer!” has become their popular rallying cry – and for good reason. Time and again, forward-thinking companies of all sizes and industries have successfully demonstrated the power of automated functional testing to thoroughly test, rapidly develop and reduce the cost of delivering high-quality applications. Yet 85% of organizations that attempt automation fail.[1] This high failure rate is symptomatic of an organizational mindset that views automation as a quick, turnkey solution. These organizations mistakenly believe that simply buying a test automation product will solve their lack-of-time and lack-of-resource problems.

There is no question that test automation can allow organizations to produce higher-quality software while leveraging existing resources. However, successful organizations realize that an automated functional testing solution is only part of the answer. To optimize software quality, speed time-to-market and heighten ROI, these companies think beyond an automated tool. They view automated testing as a much larger strategic endeavor – one that calls for an initial investment in planning a quality-driven software development process, training quality-related personnel and developing the right test framework before they can achieve the benefits of automation.

This white paper provides practical insight into the lessons learned by those who have successfully automated the functional testing process. It answers questions such as:

- What is the strategic role of functional testing in today's enterprise?
- What are the value-added benefits associated with automating the functional testing process?
- What is the best approach to ensure the success of your automation effort?
- What should you look for in an automated functional testing solution?

What is functional testing?

Functional testing, or black box testing, is a quality assurance process used to verify that an application's end-user functionality (i.e., the ability to log in, complete a transaction, etc.) works accurately, reliably, predictably and securely. Functional testing can involve either manual or automated methods. Either way, it entails running a series of tests that emulate the interaction between the user and the application in order to verify whether or not the application does what it was designed to do.

Why automate functional testing?

While manual testing is appropriate in some cases, it is a time-consuming and tedious process that is inefficient and conflicts with today's shorter application development cycles. As a result, it impedes the ability to thoroughly test an application – enabling critical bugs to slip through undetected. And if an application needs to run on multiple platforms, the manual testing effort grows in parallel with the number of platforms to be tested. What's more, manual tests are prone to human error and inconsistencies that can skew test results.

Automated testing creates new efficiencies that accelerate the testing cycle and promote software quality. By automating regression tests and other repetitive tasks, for example, QA personnel are free to expand their quality efforts. That means they can increase test coverage by extending automation to parts of the application that may not have been thoroughly tested in prior releases.

Automated testing further optimizes software quality and testing efficiency by delivering the following long-term advantages:

- **Reusability** – Automated testing enables QA to meet tight release schedules by reusing existing tests instead of starting from scratch with each new testing effort. Experience proves that reusable tests are run more frequently, enabling personnel to find and fix more errors earlier in the development process. This reusability benefit also enables QA to build libraries of repeatable test assets – in effect, transforming each test into intellectual property with long-term value.
- **Predictability & Consistency** – With automation, QA can rerun a test with the utmost consistency. This is especially critical when development creates a new build. By running regression tests, QA can quickly verify that all pre-existing functionality still works in the new version and provide early feedback to development. Consistency also applies to the testing process itself. With a repeatable process for documenting test results, QA can reproduce and verify errors – speeding the resolution process.
- **Productivity** – Automated testing creates a high-productivity environment in which organizations can increase testing capacity without additional resources. With automation, for example, QA organizations can run tests unattended on a 24/7 basis, and test an application across multiple platforms, browsers and environments simultaneously. This frees up personnel to concentrate on other quality issues. The resulting productivity gains have the dual effect of shortening test cycles and increasing opportunities to optimize software quality.

Collectively these automation advantages enable QA to accurately assess quality levels, make sound decisions regarding release readiness and minimize deployment risk.

How do you successfully automate the functional testing process?

1. Determine what applications should be automated

The best candidates for automation are enterprise applications that will require multiple releases throughout their useful life due to new, expanded or changed functionality. Applications that must produce a consistent set of results using relatively stable data are also well-suited for automation. These application characteristics enable the quality effort to fully leverage the reusability and predictability benefits of automated functional testing.

2. Choose your automated testing approach

Several methodologies exist for creating automated functional tests.

- Test modularity is an approach that divides the application under test into script components or modules. Using the scripting language of the automated testing solution, QA builds an abstraction layer in front of each component – in effect, hiding it from the rest of the application. This improves the maintainability and scalability of automated test suites by protecting the rest of the application from changes within an individual component.
- Test library architecture is another scripting-based framework that divides an application into modules which are used to build tests. Unlike test modularity, however, the test library architecture framework describes these modules in procedures and functions rather than scripts – enabling even greater modularity, maintainability and reusability.
- Keyword-driven testing (also called table-driven testing) is an application-independent framework that uses data tables and easy-to-understand “keywords” to describe the actions to be performed on the application under test. The data tables and keywords are independent of both the automated testing solution that executes them and the test scripts that drive the application and its data. Keyword-driven testing can make domain experts part of the testing process. That’s because it enables non-technical personnel to create automated tests merely by populating a simple grid with familiar terms –eliminating the need for scripting or programming. Getting started requires time to set up the keyword-driven tables and vocabulary, as well as the script components or modules behind the keywords. Once in place, however, keyword-driven testing has proven to be highly efficient and effective.
- Data-driven testing is a testing framework that stores data in an external file (such as a spreadsheet) instead of hard-coding data into test scripts. With this approach, a single script can test all the desired data values. Because the data is external to the script, this framework insulates the test script from any changes in the data. Updating the data merely requires changing the table rather than maintaining the script. This approach increases the productivity of test engineers by simplifying test maintenance and enabling a high level of script reusability from release to release. Data-driven testing is often implemented in conjunction with any of the above frameworks.
- Record/playback testing eliminates the need for scripting in order to capture a test. It starts by recording the inputs of manual interactions with the application under test. These recorded inputs are used to generate automated test scripts that can be replayed and executed later. While record/playback is the fastest and easiest approach to test automation, the tests are neither maintainable nor reusable. Any change in the application means re-recording the entire sequence of steps – negating any timesavings. The net result? Record/playback produces the lowest ROI, test asset reusability and resource productivity of any form of automation.

3. Develop your application test plan

The application test plan is a document that describes the scope, approach, resources, coverage and schedule of all the automated and manual activities involved in testing an application. Specifically, the test plan identifies all application features to be tested, the testing tasks, the individuals responsible for performing each task, the test environment, the chosen approach to test design, the various platforms and environments to be tested, and the test metrics for results reporting.

4. Create and deploy your automated tests

The application test plan serves as the roadmap for creating automated tests. Based on this plan, QA develops each test using an automated testing solution that supports the chosen approach to automation (i.e., test modularity, test library architecture, keyword-driven testing, data-driven testing or record/playback). Once the tests are mapped to requirements, defined and created, the automation is ready to start working for you.

What are the best practices of functional test automation?

Treat your automation project as a software development effort

Just like software development, test automation must be carefully designed, documented and reviewed. Time and again, history proves that seat-of-the-pants implementations are doomed to failure.

Successful organizations view quality holistically across the entire software application lifecycle and apply best practices when implementing test automation. For example, they move testing up early in the development process – even as early as requirements definition. They use modular code that is easy to maintain. In addition, they build test assets that can be reused in later quality phases and future versions of the application.

Implementing test automation is not a short-term project. Those who succeed understand that test automation is a full-time effort that requires a significant upfront investment in time and resources. So they set management's expectations accordingly. They also realize that an automated solution is no substitute for expertise. So they invest in the appropriate training and develop the right skill set by learning about best practices for automation, testing methodologies and the test-automation solution itself.

Ensure collaboration to optimize software quality

Quality is not just the domain of the QA tester; it is a goal that must be owned by the entire company. By creating a collaborative environment with domain experts and software developers during pre-deployment and with operations throughout post-deployment, QA can capture all the relevant input needed to optimize software quality and ensure that applications meet business requirements.

One way in which successful QA organizations promote such collaboration is by creating the position of Automation Architect to fill the role of a test developer. The Automation Architect collaborates with domain experts to understand what business functions the application must perform and how the application must work in the eyes of its users. The Automation Architect can then design the test automation framework accordingly, ensuring that the application will meet its business requirements and end-user needs for functionality, quality and reliability.

Another way to foster collaboration is to seek a software test automation vendor with an integrated platform that expands the quality process beyond functional testing to encompass the entire software application lifecycle. Such a platform infuses

quality each step of the way – from initial design, to deployment and into production. It also promotes cross-functional communication by enabling all stakeholders to access and share the quality-related information captured within the platform repository.

Manage constantly changing applications

Change is inevitable – and extensive – in software applications as new business objectives, user requirements and computing environments emerge. In fact, the software maintenance required to keep pace with changing applications accounts for more than 70% of all development costs.[2]

Successful companies mitigate these costly maintenance activities by building maintainability into test automation. For example, they create modular test components that isolate and encapsulate application functionality. When a change in functionality occurs, they only have to modify the effected component(s) instead of rescripting the entire test – enabling them to quickly and easily modify tests to reflect changing application needs.

Ease of maintenance and reusability are two major areas in which test automation drives ROI. The best way to build these two attributes into test automation is to adopt an automation solution with an object-oriented architecture. Such an architecture allows the use of any chosen approach to automation (i.e., test modularity, test library architecture, keyword-driven testing, data-driven testing or record/playback). Most importantly, an object-oriented architecture provides a dramatic reduction in the time and effort necessary to modify tests in order to reflect new or expanded functionality. As a result, organizations can keep up with application changes and maintain a high level of reusability among test assets from build to build.

What should you look for in an automated functional testing solution?

A test automation environment designed for maximum productivity – To keep pace with today's short product release cycles, QA organizations need an automated solution that enables them to build modular and reusable test assets. In addition to accelerating the testing process, reusable test cases optimize software quality by increasing the predictability of test runs.

High adaptability to cope with ever-changing business requirements – New business requirements can emerge literally overnight – creating an immediate need to revamp an application. But will you be able to revamp all of the associated test cases in time? Look for an automated solution that minimizes the burden of heavy maintenance. The inherent reusability and easy maintenance of an object-oriented testing solution will provide the ability to change test scripts quickly, along with the portability and extensibility to embrace different technologies.

Broad technology support – Today's enterprise applications must support users across a wide diversity of platforms, environments and technologies. To avoid the need to purchase multiple testing products, select an automated solution that provides an all-inclusive testing environment to support all system configurations.

Quality optimization throughout the application lifecycle – Though functional testing may be an initial priority, quality does not begin and end with functional testing. Ultimately, quality needs to extend to other areas such as unit testing, performance and load testing, test management and application performance management once the application is in production.

Established community of users – In today’s technology-driven business world, enterprise applications can make or break corporate success. Mitigate the risk of application failure by seeking a proven solution whose success has been demonstrated within a wide community of users.

Services and support to meet user needs – Implementing test automation initially requires time and resources – both of which are often constrained in today’s software development environments. Choose an automated testing solution from a vendor capable of providing value-added consulting and training services to supplement your in-house resources and get your automated testing environment up-and-running successfully.

Borland® SilkTest® – the automated functional testing solution

Borland® SilkTest® is the industry-leading, award winning automation solution for testing the functionality of all enterprise applications, whether .NET, Web, Java™ or client/server-based. SilkTest enables companies to verify that enterprise applications reliably meet business goals within the constraints of today’s testing cycles by leveraging the accuracy, consistency and timesaving benefits of its powerful automated testing technology. SilkTest also supports all major testing frameworks including test modularity, test library architecture, keyword-driven, data-driven or record/playback. In addition, SilkTest ensures the reliability of multilingual applications (including double-byte character sets such as Kanji or Simplified Chinese) by providing the industry’s most powerful Unicode®-enabled functional testing solution.

Designed for productivity and ease of use, SilkTest maximizes the speed and efficiency of the testing process. For example, users can test across multiple platforms, browsers and technologies with one test script by capitalizing on SilkTest’s flexible and robust object-oriented scripting language (Borland® 4Test®). What’s more, SilkTest’s built-in, customizable recovery system lets users run tests unattended on a 24/7 basis. With less time spent on repetitive testing tasks, QA can expand test coverage to catch more errors earlier in the development process and optimize application quality.

In addition to validating an application’s functionality prior to its initial release, SilkTest users can evaluate the impact of new features on existing functionality by simply reusing existing test cases. These reusable test assets minimize maintenance by letting users modify tests quickly and keep pace with application changes.

The product architecture of Silktest enables organizations to cost-effectively manage the complexities of testing enterprise applications. Its unique Agent technology is instrumental in providing the control and granularity required to test intricate, real-life business scenarios. In addition, SilkTest Runtime, which is a scaled-down version of SilkTest, lets organizations distribute tests to multiple machines and environments – increasing the productivity while decreasing the cost of testing activities.

Lifecycle quality management

The Borland Lifecycle Quality Management Solution enables companies to deploy high-quality software, reduce business risk and maximize return-on-investment. Leading businesses around the world, including many of the Fortune 500, rely on Borland's innovative Silk family of products to protect their business service levels, competitive edge and brand reputation. And because quality is not solely about implementing a tool, Borland offers a full range of consulting, training and support services – all designed to ensure the success of each customer's automated testing environment.

In addition to SilkTest, Borland offers:

- **Borland® SilkCentral® Test Manager** for managing the entire testing process.
- **Borland® SilkPerformer®** for enterprise-level load and performance testing.
- **Borland® SilkPerformer® Component Test Edition** for early testing of remote components under concurrent access.
- **Borland® SilkCentral® Performance Manager** for monitoring applications from the end-user perspective.

Summary

With the right planning and effort, automated functional testing can optimize software quality by verifying the accuracy and reliability of an application's end-user functionality in pre-production. Automated testing also creates new efficiencies which ensure that even complex enterprise applications are deployed on time and on budget. Today, companies around the world are achieving these benefits with SilkTest, an award-winning automated functional testing solution from Borland. SilkTest provides a repeatable, predictable method of testing the functionality of enterprise applications from build to build, over time. As a result, SilkTest builds speed and accuracy into the entire testing process and optimizes quality by freeing up QA to expand test coverage and proactively catch errors earlier in the development process.

References

- [1] International Institute for Software Testing
- [2] Guelph Natural Computational Research Group, University of Guelph