HAPPY ABOUT®

GLOBAL SOFTWARE TEST AUTOMATION

A Discussion of Software Testing for Executives

BY HUNG Q. NGUYEN MICHAEL HACKETT BRENT K. WHITLOCK

"Software is complex but I'm tired of finding bug after bug that a 5th grader wouldn't have turned in. Virtually every technical product these days includes a lot of software. It's a rare engineer that can write nearly perfect code. Methodical and thorough testing of software is the key to quality products that do what the user expects. Read this book to learn what you need to do!"
Steve Wozniak, Wheels of Zeus, CTO

HappyAbout.info

“Happy About® Global Software Test Automation” Book Excerpt

A Discussion of Software Testing for Executives

By Hung Q. Nguyen, Michael Hackett and Brent K. Whitlock

Subset of the book brought to you by Happy About

20660 Stevens Creek Blvd.
Suite 210
Cupertino, CA 95014

WHITE PAPER Table of Contents (included here)

Getting "Happy About Global Software Test Automation" (http://www.happyabout.info/globalswtestautomation.php)

"Happy About Global Software Test Automation" can be purchased as an eBook for $11.95 or tradebook for $19.95 at: http://www.happyabout.info/globalswtestautomation.php or at other online and physical book stores.

Please contact us for quantity discounts sales@happyabout.info or to be informed about upcoming titles bookupdate@happyabout.info or phone (408-257-3000).

Contents

NOTE: This is the Table of Contents (TOC) from the book for your reference. The eBook TOC (below) differs in page count from the tradebook TOC.

Chapter

3 The Pitfalls of Manual Software Testing

In this chapter, we will present the top five pitfalls of manual software testing, listed below:

1. Manual testing is slow and costly.
2. Manual tests don't scale well.
3. Manual testing is not consistent or repeatable.
4. Lack of training.
5. Testing is difficult to manage.

We will then present the top five suggestions, listed below:

1. Be thorough in test design and documentation.
2. Automate the turnkey tests as much as possible.
3. Manage the test activities well.
4. Rank test cases in order of importance.
5. Have a separate budget with proper funding for testing.

We will then present some case studies that illustrate the issues.

Introduction

Manual software testing has been the cornerstone of software testing. All test engineers and software QA staff, software engineers, developers, and programmers test their code manually, at least to some degree. Manual software testing is employed with all sizes of projects and budgets, ranging from zero to billions of dollars. It is a critical element of software testing, but not the be-all and end-all. Furthermore, it is not synonymous with quality assurance. Just as in software development, the quality of the results of manual software testing can vary widely depending on many factors.

The Top Five Pitfalls of Manual Software Testing

Manual software testing is a necessity, and an unavoidable part of the software product development process. How much testing you do manually, as compared to using test automation, can make the difference between a project's success and failure. We will discuss test automation in more detail in a later chapter, but the top five pitfalls of manual software testing illuminate areas where improvements can be made. The pitfalls are listed and described below:

1. **Manual testing is slow and costly**. Because it is very labor-intensive, it takes a long time to complete tests. To try to accelerate testing, you may increase the headcount of the test organization. This increases the labor as well as the communication costs.

2. **Manual tests don't scale well**. As the complexity of the software increases, the complexity of the testing problem grows exponentially. If tests are detailed and must be performed manually, performing them can take quite a bit of time and effort. This leads to an increase in the total time devoted to testing as well as the total cost of testing. Even with these increases in the time and cost, the test coverage goes down as the complexity goes up because of the exponential growth rate.

3. **Manual testing is not consistent or repeatable**. Variations in how the tests are performed are inevitable, for various reasons. One tester may approach and perform a certain test differently from another, resulting in different results on the same test, because the tests are not being performed identically. As another example, if there are differences in the location a mouse is pointed when its button is clicked, or how fast operations are performed, these could potentially produce different results.

4. **Lack of training is a common problem, although not unique to manual software testing**. The staff should be well-trained in the different phases of software testing:

   - Test design
   - Test execution
   - Test result evaluation

5. **Testing is difficult to manage**. There are more unknowns and greater uncertainty in testing than in code development. Modern software development practices are well-structured, but if you don't have sufficient structure in testing, it will be difficult to manage. Consider a case in which the development phase of a project schedule slips. Since manual software testing takes more time, more resources, and is costly, that schedule slip can be difficult to manage. A delay in getting the software to the test team on schedule can result in significant wasted resources. Manual testing, as well as badly designed automated testing, are also not agile. Therefore, changes in test focus or product requirements make these efforts even more difficult to manage.

The Top Five Suggestions

There are ways that pitfalls associated with manual software testing can be avoided or resolved. In this section, we discuss five of these.

1. **Be thorough in test design and documentation**. In designing the tests, there should be agreement among the business staff, product and project managers, developers, and testers on test coverage. This can be documented as test requirements in a test plan. With this documentation, management can have visibility

into the test coverage and know that the right areas are being tested. This then becomes an important management tool in managing testing.

The goal is to find the easiest way to document as many test cases as possible without having the test effort turn into a documentation effort.

Have the test requirements and test cases peer-reviewed, just as you would have software design reviews. The software development staff and the test staff should jointly develop the test designs, as they have complementary skill sets and knowledge bases.

In manual testing, like in other processes, several factors influence the effectiveness of the tests, including the completeness of the test cases and the thoroughness of the documentation. The goal should be to maximize management's understanding of the testing by spending the appropriate resources in each area, within the overall resource constraints. If you don't document your tests, you won't understand the coverage or software quality as revealed by the tests, and you also won't be able to determine that the test team is testing features that are the most important to your development team and customers. However, if you document everything related to each test case, you won't have time to do as many tests as you should. Documenting test cases can get expensive. The goal is to find the easiest way to document as many test cases as possible, without having the test effort turn into a documentation effort.

2. **Automate the turnkey tests as much as possible**. There are various tools available that support test automation. When it is cost-effective and time-efficient to do so, there is no excuse for not automating software tests. The benefit of automation is that the testing becomes less burdensome, and less likely to be scrimped on when under pressure. This also makes the testing easier to manage.

"Software testing is surprisingly low-tech, and requires too much 'hands-on.' Would most people run their virus software frequently if they had to invoke it manually?"
*—**Sue Kunz**, CEO of SolidWare Technologies*

3. **Manage the test activities well**. Do this as closely, and by establishing as full a procedure, as the software development activities. Be sure that the project plan has sufficient resources and time allocated to testing so that it doesn't get short shrift.

4. **Rank test cases in order of importance, by impact on quality, by risk, by how often the feature tested is used, or some other related metric**. A goal should be to run all important test cases, but if there are resource constraints that prevent all test cases from being run, then the ranking will enable the important test cases to be run. This provides the maximum impact of the testing with the available resources. In manual software testing, you are always short on time. There should be an agreement or signoff procedure on the ranking and the coverage of the tests.

5. **Have a separate budget with proper funding allocated for testing**. Just as there is a budget allocated for the software code development, there should be a budget allocated for testing. Be sure that this budget is well-funded. Watch how much is spent on testing, and what the return on investment is.

Even when you have a good test automation program in place, you will still need to do some manual testing. The usability testing, for example, requires human involvement. However, manual testing is not the solution for short-cycle, high-volume test challenges. For example, if you have a daily build process where you need to run smoke-tests to assess the changes and stability of the software from build to build, these tests can be high-volume and short-cycle. Manual testing cannot solve your testing problem in this case.

Case Studies

We surveyed CxO's and senior managers in a wide range of software companies, inquiring about their views and war stories related to manual software testing. Responses and suggestions we received on this subject appear in the following paragraphs. We are sharing with you the benefit of learning from their experience.

Clear test planning, coupled with well-defined acceptance tests at all levels, provide visibility into the management of customer expectations.

Michael Abbott, *CTO and founder of Composite Software*, shared with us the following:

> "Software testing in the U.S. has been traditionally not viewed as a priority for many development organizations. Typically, with ever-shrinking deadlines and a demand for increased productivity, many engineering managers cut the time for testing software in favor of fewer features and/or enhancements. This will change during this decade, as a larger focus on quality will be demanded by customers and is already being reflected in many negotiated software license agreements with *Fortune 100* companies. Companies are discovering that the operational cost for poorly-tested software is significantly affecting their bottom line."
>
> "One of the many challenges in software testing is building the test plan with the appropriate test cases to outline the focus on the quality evaluation process. Many companies fail to clearly build a strategy around testing, and thus fail to communicate to the implementation team in the field what has and has not been tested. To address this issue, acceptance criteria needs to not only be a part of a testing plan for the passing of a product from development to testing, but [is] also [essential] to professional services. As a part of the internal release notes, the test case coverage needs to be clearly communicated in order to mitigate product risks with customers in the field. This approach resulted in an increase in successful proof of concepts at the customer sites, as well as more effective management of expectations by internal and external personnel."

Consider speed versus cost in your strategy.

Clive Boulton, *Senior Developer of Exact Software*, shared the following with us:

> "Experience is a great teacher. Regarding testing ERP software, including e-Business and accounting… In the past, before global labor resources [were available], we automated to save money on-shore, but found manual testing was more essential and flexible but so costly. We recommend automatic testing on-shore to cut costs, manual testing off-shore where labor costs are not critical. The hybrid of both automated and manual testing are required. Keep in mind, automating testing before a product is version 1.0 is just not practical."

Manual testing, while not scalable, is often unavoidable.

Mark Tezak, *Acrobat Quality Engineering at Adobe Systems, Inc.*, shared with us the following about the problems and benefits of manual software testing:

> "As the areas I test concern document creation for print production, there is not a lot of automation I employ at all, save for some very basic scripting of repetitive tasks. Much of this functionality is built into the programs, as well for the user, in the form of batch processing commands. So the expertise I bring to the task is two-fold, namely workflow production knowledge resulting from seventeen years of experience of the publishing and printing industry, combined with the skills I have learned in software quality engineering. As a result, I am able to not only create real-world user scenarios and workflows, but also to quickly identify a chain of defects easily once a vulnerability has been discovered in a particular area during product development, since I am aware of the functional interdependencies."

Poor funding and lifecycle management leave little room for testing improvement.

Shyamsundar Eranky of *Symbol* shared with us the following:

> "I feel that QA is perhaps the most important part of software development and sadly the most neglected—either due to a time or resource crunch. On manual software testing, even though this is not the most reliable form of testing, sometimes it is the only form of testing available. This was true more so for projects that involved GUI testing than server side testing. Although software automation tools exist for such testing, it is often not suited given the time to complete the QA."

Summary

Manual software testing is slow and costly, does not scale well as the complexity of the software increases, and has lack of repeatability and consistency in results. To improve software testing, you should automate the testing process as much as you can, while allocating sufficient resources for manual testing. Providing the test function with its own budget, as opposed to a flexible portion of the overall project budget, is very useful here. Managing the software test process with as much care as the software development process, from documentation of the test plans and ranking of the tests to test execution and reporting, will also be helpful.

Driving toward the solution

In the last chapter, we discussed:

- The Pitfalls of Manual Software Testing.

In the following two chapters, we will discuss:

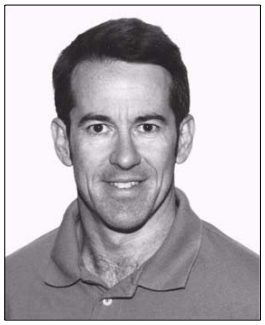
- The Pitfalls of Software Test Automation
- The Pitfalls of Outsourcing/Offshoring Software Testing

In Chapter 6, we will present the Global Test Automation strategy, an approach that avoids these pitfalls and enables you to capitalize on the value that software testing can provide.

About the Authors

Hung Q. Nguyen is Founder and CEO of LogiGear, responsible for the company's strategic direction and executive management. He's been a leading innovator in software testing, global test automation, testing tool solutions and testing education programs for over two decades. Nguyen and LogiGear have helped companies, from Fortune 500 to startups, delivering unique testing solutions which double their test coverage, cut test time in half, improve quality and reduce cost. As one of the top thought-leaders in the software testing industry, Nguyen is coauthor of the top-selling book in the software testing field, Testing Computer Software (Wiley, 2nd ed. 2002) and other publications including Testing Applications on the Web (Wiley, 2nd ed. 2003). Nguyen's experience includes leadership roles in software development, quality, and product management at leading software companies.

Nguyen is also a director of two non-profit organizations, the Association for Software Testing, an organization dedicated to improving the practice of software testing by advancing the science of testing and its application which he co-founded, and San Francisco Bay Jazz Ensemble, an eighteen-piece big band with a mission to provide live music to local community as well as through international venues to further public awareness and appreciation of the valuable American musical heritage, jazz. He holds a Bachelor of Science in Quality Assurance from Cogswell Polytechnical College.



Michael Hackett co-founded LogiGear in 1994 and leads the company's LogiGear University training operations division, setting the standard in software testing education programs for many of the world's leading software development organizations. Mr. Hackett is coauthor of the popular Testing Applications on the Web (Wiley, 2nd ed. 2003), and has helped many clients produce, test and deploy applications ranging from business productivity to educational multimedia across multiple platforms and multiple language editions. His clients have included Palm Computing, Oracle, CNET, Roche Pharmaceuticals, Pfizer and Bank of America.

He is on the Board of Advisors for the Software Quality Engineering and Management Certificate Program at University of California at Santa Cruz. Michael's training has brought Silicon Valley Quality and Testing Expertise to 10 countries around the world. He holds a Bachelor of Science in Engineering from Carnegie-Mellon University.

Brent K. Whitlock is currently with the patent practice group at Carr & Ferrell LLP where he helps inventors, entrepreneurs, and executives develop their intellectual property strategies and patent portfolios. He is also a founding shareholder of RSoft Design Group, Inc., where as Director of Optical Systems Research and Business Development, he initiated and led the development and commercialization of several optical communication system simulation software packages including LinkSIM™, ModeSYS™, and OptSim™ 4, which won the Lightwave OFC/NFOEC 2005 Attendees Choice Award. He has also secured and served as Principal Investigator on federally funded SBIR, STTR, and NIST ATP research contracts.

Dr. Whitlock earned his B.S., M.S., and Ph.D. all in Electrical Engineering from the University of Illinois at Urbana-Champaign. Dr. Whitlock has co-authored over 30 technical papers and articles. He is a Sr. Member of the IEEE and Chair of the Santa Clara Valley chapter of IEEE LEOS.

About LogiGear®

LogiGear is the leading provider of global solutions for software testing, focusing on test automation. Founded in 1994, led by top thought leaders in the software testing industry, and supported by a bright, hard-working staff that has a passion for software testing, and that all-important eye for detail, LogiGear has provided effective software quality solutions to clients ranging from the Fortune 500 to early-stage startups. LogiGear works closely with its customers to determine their exact software quality testing goals and challenges, then designs unique solutions based on our onshore/offshore testing services, test automation tools, QA training, and consulting.

LogiGear partners with its customers to ensure that they have the right approach for test automation success. Based on the unique goals and needs of an organization, LogiGear provides varying solutions such as a complete Global Test Automation solution, Action Based Testing™ training, consulting, coaching, TestArchitect tooling or integration of existing third-party or homegrown tools into the TestArchitect framework.

LogiGear is a privately funded corporation headquartered in Foster City, California.

About TestArchitect™

TestArchitect provides a powerful test automation framework supporting the Action Based Testing™ method, allowing software development teams to double their test coverage and decrease testing time, leading to better product quality and reduced costs.

TestArchitect is flexible and easy-to-use, and draws on LogiGear's many years of experience in the software quality assurance industry, including the leadership of the original architect of the keyword driven testing method. LogiGear offers turnkey test automation solutions that will benefit the entire software quality process, from test design to team management.

Reducing time-to-market and improving product quality are critical to the success of any software organization. TestArchitect enables all members of the team to improve the testing effort. Testers and business analysts have an easy-to-use method for creating intelligent, maintainable tests which can be executed manually or automatically. Automation engineers have powerful tools based on industry-standard languages for creating the underlying test automation. Managers can maintain control and efficiency of their global testing efforts through a central repository and easy-to-use, customizable reports. Business stakeholders can reduce testing cost through our proven ROI model.

A Message From Happy About®

Thank you for your purchase of this Happy About book. It is available online at: http://happyabout.info/globalswtestautomation.php or at other online and physical bookstores.

- Please contact us for quantity discounts at sales@happyabout.info
- If you want to be informed by e-mail of upcoming Happy About® books, please e-mail bookupdate@happyabout.info
- If you want to contribute to upcoming Happy About® books, please go to http://happyabout.info/contribute/
- Please see our web page http://www.happyabout.info/globalswtestautomation.php

Happy About is interested in you if you are an author that would like to submit a non-fiction book proposal or a corporation that would like to have a book written for you. Please contact us by e-mail editorial@happyabout.info or phone (1-408-257-3000).

Other Happy About books available include:

- Happy About Joint-Venturing: http://happyabout.info/jointventuring.php
- Happy About LinkedIn for Recruiting: http://happyabout.info/linkedin4recruiting.php
- Happy About Website Payments with PayPal: http://happyabout.info/paypal.php
- Happy About Outsourcing: http://happyabout.info/outsourcing.php
- Happy About Knowing What to Expect in 2006: http://happyabout.info/2006economy.php

Other soon-to-be-released Happy About books include:

- Happy About CEO Excellence: http://happyabout.info/ceo-excellence.php
- Happy About Working To Stay Young: http://happyabout.info/working-to-stay-young.php
- Happy About Open Source: http://happyabout.info/opensource.php