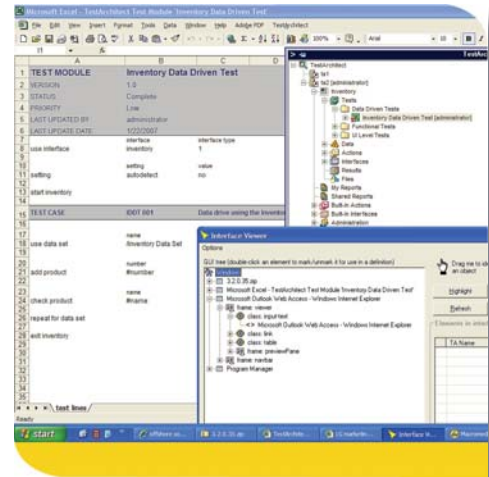


Synopsis

This paper traces the evolution of software testing through its first two phases, and then presents the current state-of-the-practice: Software Testing 3.0, an independent and strategic approach to software quality.

The paper concludes with a discussion for senior executives on how and why software testing can finally meet and exceed management expectations, illustrated by real-world case studies showing how two companies benefited from Software Testing 3.0 to test better and faster while lowering their costs.



Contents

Introduction.....	1
The Early “Evolution” of Software Testing	1
“Disconnects” in Software Testing.....	2
The Evolution to Software Testing 3.0.....	3
The Executive Perspective on Software Testing 3.0.....	5
Why Treat Testing as a Strategic Initiative?	5
Why Should Testing be its Own Organization with a Separate Budget?	5
Why is Visibility so Important?	6
Case Studies	7
1. Security software - <i>Automation Expands Test Coverage, Reduces Test Time</i> ...	7
2. Networking software - <i>Testing Mature Software that was Never Tested</i>	9
Conclusion.....	10
LogiGear Contact Information.....	10

Introduction

Software testing has undergone an evolutionary process. The early stages of software testing were primitive and did not fully meet intended goals: better quality products through broader and deeper testing coverage, taking less time to test, and reducing quality-related costs.

The current state-of-the-practice, which we refer to as “Software Testing 3.0”, is now emerging. Software Testing 3.0 offers the culture, methodologies, beliefs, and techniques that will enable software testing to truly deliver the full potential of quality engineering, test automation and global resources.

This paper will trace the evolution of software testing through its first two phases, discussing the shortcomings or “disconnects.” It will then present Software Testing 3.0, and conclude with a discussion of why senior executives should care – in essence, why it matters to them that software testing is finally evolving to a point where it can meet and exceed management expectations.

The Early “Evolution” of Software Testing

Software testing has already gone through at least two clearly identifiable phases. In “Software Testing 1.0”, software testing was misunderstood. It was an afterthought or adjunct to the development process left to less skilled, lower-paid individuals who actually viewed software testing as the entry point on a career path to becoming a software developer. There were few useful testing tools and methods for achieving a high degree of automation. Those that existed were expensive, complex, difficult to use, and ineffective at addressing productivity needs and concerns. In this phase, executive management was mostly disconnected from software testing – assuming that somehow, it would get done.

With “Software Testing 2.0” came the recognition that software testing was and is an important part of the development process. Software testing was recognized as a valuable part of application development, and everybody started to test in earnest. With the high degree of acceptance across functional groups, came ownership issues – issues of where testing fits organizationally, how it is budgeted for, and from where testing should receive its direction. There was also an explosion of available tools. These tools often further distracted the testing efforts, turning the process into one of tool selection without clear goals, architecture or direction. Within this phase, executive management’s understanding of testing was still rudimentary as was their involvement.

“Disconnects” in Software Testing

Both of these phases underperformed from the standpoint of organizations’ expectations for testing. There are some significant common threads throughout these first two phases that are at the root of our unmet expectations. The first is the lack of involvement and understanding on the part of senior management. Without a clear understanding of the importance of testing, its place within the development process and its valuable contributions to the organization, testing hopes can be dashed upon the rocky shores of misguided direction, funding, and expectations.

The second thread that has caused significant problems is the emphasis on testing for testing’s sake, and tools and tool selection without any strategy for guidance. This is like building a road without destinations. You will end up with a road, but its ability to meet anyone’s needs is left entirely to chance. Similarly, testing and tool selection without a strategy will provide an organization with a software testing program, but the chances that it will provide better quality products through better testing coverage, in less time and at less expense, are very slim indeed. In essence, the first two phases yielded testing programs that were ill-equipped to actually meet the needs of the organization for better software made faster and cheaper.

These “disconnects” yield complaints that are all too common for 1.0/2.0 software testing programs, such as:

- “Our organization struggles with test automation and offshoring and we do not see significant gains in coverage or quality, or reduced costs...”
- “Software testing requires too much hands-on...”

The Software Testing Business “by the Numbers”

Through the first two phases of software testing’s evolution; software testing has evolved into a multi-billion dollar industry.

- The global software testing market is \$13 billion (IDC¹)
- The size of the outsourced testing services market is approximately \$6.1 billion (Dataquest)
- The market for automated software quality tools for distributed environments is approximately \$1.2 billion, with a compound annual growth rate of 14.9% (IDC²)
- Software development is up to 40% of a typical product release budget, and software testing is 40% of the development budget (LogiGear internal studies³)
- Offshore software testing, application maintenance, software deployment and support services are an over \$3.4 billion business (IDC¹)
- The national cost estimate of an inadequate infrastructure for software testing is \$59.5 billion (NIST⁴) due to:
 - Inadequate integration and interoperability testing
 - Inadequate automated generation of test code
 - Lack of a rigorous method for determining when a product is ready for release
 - Lack of performance metrics and testing measuring procedures
- The potential cost reduction from feasible infrastructure improvements is \$22.2 billion (NIST⁴)

- “Software quality problems are as pervasive as ever in spite of advancements in development techniques and technologies...”

As shown in the sidebar “The Software Testing Business by the Numbers,” testing is emerging into a healthy industry and an established discipline. Yet, although executive management has a much clearer vision of the marketing, sales and finance parts of the product release process, most executives still lack an understanding of the challenges of quality and software testing, or they understand the challenges but have little confidence in the proposed solutions. This is exacerbated by having little or no visibility into the effectiveness – or rather, the ineffectiveness – of testing. As a result, offshoring of software testing has been unsatisfactory due to problematic communications, insufficient or mismatched skill sets at the outsourced software test organization, and problems with management, vendor relations and global infrastructure. Furthermore, test automation implementation has been failing due to many pitfalls, including uncertain return-on-investment, lack of visibility, and poor reusability, scalability and maintainability.

Management needs to have a clearer understanding of the challenges of software testing and greater confidence in proposed solutions. They need visibility into testing and its effectiveness or lack thereof. These are the challenges that are met by Software Testing 3.0.

The Evolution to Software Testing 3.0

Software Testing 3.0 starts with the understanding that testing is still valuable to the organization. The foundation for Software Testing 3.0 is a strategic end-to-end framework for change based upon a strategy to drive testing activities, tool selection, and people development. Fundamental to its success is the realization that testing is its own function, and an integral part of the development process. Testing must have its own budget to build the necessary infrastructure, as well as clear leadership and ownership. It must provide executive management with visibility into the status of the software under test so that executives can make intelligent and informed decisions about projects and have confidence that they are developing and delivering a quality product that will meet customer needs.

SP3™:

The Components of an Effective Test Effort

A test strategy has three components that need to work together to produce an effective test effort. LogiGear has developed a model called SP3, based on a framework developed by Mitchell Levy of the Value Framework Institute. The components of this strategy (S) consist of:

1. People (P1) - everyone on your team
2. Process (P2) - the software development and test process
3. Practice (P3) - the methods and tools your team employs to accomplish the testing task

Moving from Software Testing 2.0 to Software Testing 3.0 requires a strategic commitment on the part of the organization, and elevation of software testing from a task or tactical effort to a “strategic effort”. An organization must form a team or task force to address the needs of people, practice, and process (SP3™). This task force will develop a team-based methodology to drive tool selection (tools that support the methodology) as well as people development (training in the methodology, tools, and software testing techniques that support the methodology), and determination of necessary metrics to provide management with visibility into the status and quality of the software under test.

This transformation of culture, methodology, beliefs, and techniques will yield a team-based software testing program that is test design focused with an emphasis on testing as a whole rather than in parts. It will address feature reliability; compatibility, usability, and interoperability; performance acceptability, availability, and scalability; security; as well as accessibility. More importantly, it will yield a software testing program that:

- improves quality by increasing testing coverage and effectiveness
- decreases testing time
- decreases costs
- provides the necessary information for informed decision making
- provides several assets, including:
 - tested, high quality software
 - reusable software tests that codify organization knowledge about the software under test and provide the basis for the testing of future generations of the software
- focuses on test design and agile test development practices, and
- extends test automation to multiple platforms including non-GUI platforms

Software Testing 3.0 In Action

Employing Software Testing 3.0, LogiGear has helped a large oil and gas exploration company to:

- change the methodology they use for testing, enabling them to become more efficient
- automate tests that had traditionally been done manually,
- integrate the cost-benefits of onshore and offshore testing, and
- run more tests - and better tests - in the available cycle time

In addition to benefiting from implementation of Software Testing 3.0 principals, with LogiGear’s involvement the company has solved some particularly difficult testing problems that had been plaguing them for some time.

The Executive Perspective on Software Testing 3.0

How can an organization afford the effort to move to Software Testing 3.0? Actually, the more important question is “How can an organization afford not to move to Software Testing 3.0?”

Software Testing 3.0, treating software testing as a strategic effort, enables an organization to turn quality into a strategic advantage. To be convinced of the value of this proposition, it is important for senior management to understand the “whys” of Software Testing 3.0.

Test Faster

Employing Software Testing 3.0 principals, LogiGear provided a mobile wireless services company with a global, maintainable and scalable automated testing framework that completely tests their system in a matter of days rather than weeks.

Software Testing 3.0 principals also enhanced their overall testing effort through more effective test planning and manual testing.

Why Treat Software Testing as a Strategic Initiative?

In our experience, software development represents 40% of the product release budget for any company that develops software for resale. Software testing typically represents 40% of the development budget. Any company task that represents that much money should be treated as a strategic initiative. Strategically “doing-the-right-thing” coupled with “doing it right” can have a profound impact both on the outcome as well as the cost. Conversely, treating it as a tactical initiative can (and will) yield sub-optimal results and lead to unnecessary costs. Quite simply, Software Testing 3.0 will save time and money, and result in higher quality software.

Why Should Software Testing be its Own Organization with a Separate Budget?

There are several reasons why software testing should be a separate organization from development, with its own budget. First, it is important to realize that software testing performs functions that are very different from software development. Development is a creative process of providing new functionality and features. Testing is a destructive process where testers actually try to “break” the software so that customers do not.

Testers are not developers. The skills and experience needed differ from those of a developer. Having testing as its own organization with its own budget allows testing-centric management to nurture and develop testing-specific skills in their testers (skills in methodology, tools, metric reporting, etc.).

Having testing as its own organization also gives it the independence necessary to “stand up” to

development when there are problems with the software under test. It is much less likely that development will be able to ram sub-standard code through an independent testing organization to meet delivery promises without a red flag being waved to senior management. While this may potentially cause delays in delivery, these are delays that can prevent significant downstream costs as well as lost revenue and customer goodwill that results when bad software makes it to market. Having software testing and its budget separate prevents testing getting short changed or circumvented and will help to deliver higher quality products.

Software Testing's budget should also be separate because there are large amounts of money involved. Having the budget separate gives the organization the ability to exert more direct fiscal control and oversight. With an independent budget for software testing, it becomes much easier to measure the return and the costs savings that may be realized from initiatives such as automation or offshoring of testing efforts.

Test Better and Reduce Costs

With Software Testing 3.0, LogiGear helped a provider of software solutions for the communications and media industries to double their testing coverage, run an expanded test suite 500% faster, and reduce overall testing and test automation costs by 31%. LogiGear's implementation of Software Testing 3.0 allowed them to increase test coverage and the quality of each release, while reducing costs.

Why is Visibility so Important?

Software Testing performs several key functions for the benefit of the company, product development, and company management. Testing is a service to the company that helps the organization produce and release higher quality software. Testing is also a service to the development team to help it produce higher quality code. For senior management, testing is an information service providing information about product quality.

Testing performs these services by providing key metrics or data, including test cases run, testing coverage statistics, defect reports and more. This data provides visibility into the status and quality of the software under test. This data can be used by the testing group itself to further refine and enhance testing efforts. Development can use this data to gain an understanding of what parts of the product are causing problems, which may lead to new processes or development tactics to deliver better code to testing. Senior management uses this data to make critical decisions about product schedules and delivery, as well as resource allocations.

With visibility comes:

- Organizational improvement
- Better confidence in the quality (consistency and dependability) of the software
- Better decision making, and better confidence in the decision making because it is based on data

- Fewer “surprises”
- Better allocation and utilization of resources and budget
- Ability to spend more time and money on development and less on maintenance
- Delivering higher quality software

Case Studies

LogiGear has extensive experience implementing Software Testing 3.0 concepts in clients of many sizes across many industries. In addition, LogiGear’s executive team are recognized industry thought leaders on software testing, automated software testing, and offshoring of software testing – all based on a foundation of Software Testing 3.0 concepts.

The experience of the following companies is typical of LogiGear-provided solutions based upon Software Testing 3.0.

Software Testing 3.0 Case Study #1

MX Logic Inc.: Automation Expands Testing Coverage, Reduces Testing Time

Environment Prior to LogiGear:

MX Logic is a leading provider of email and Web managed security services. The company was running a suite of 423 software tests manually to test their security software. Manual testing was so time consuming that:

- Manual testing of their software’s functionality was taking approximately 240 person-hours to run
- Build acceptance testing was done on an ad-hoc, time-permitting basis

LogiGear’s Solution:

LogiGear implemented an automated testing program for MX Logic that made use of:

- Low-cost off-shore LogiGear testing resources in Vietnam. These resources are all trained by LogiGear University in Software Testing 3.0 concepts and tools.
- TestArchitect™, a tool set that integrates the latest methodologies and technologies in one easy-to-use package.

The automated software testing solution implemented by LogiGear allowed the company to:

- Automate existing tests and expanded test coverage to 945 automated test cases that run in 150 to 160 machine hours (the same tests run manually would consume 480 to 500 man hours).
- Expand testing coverage from 50% to 90%
- Expand automated testing coverage from 10% to 80%
- Use a subset of these tests to implement regular automated build acceptance testing that runs in 2 machine hours (the same would take 1 person day manually)
- Reuse the automated test suite release after release with only the addition of a small number incremental test cases to test new functionality
- Avoid staff growth in their domestic operations

Using the difference in cost between fully-burdened U.S. based resources and Vietnam based resources, LogiGear estimates that this level of automated testing effort done manually would have cost the company an additional \$1 million or more. LogiGear's automated software testing efforts have expanded testing coverage and effectiveness while avoiding significant additional expense.

In addition, LogiGear created performance testing to help MX Logic monitor their customer experience on their production portal. They run this daily to ensure that their customer experience is not negatively affected by any new features or Internet traffic spikes.

Case Study #1, Next Steps:

LogiGear will be producing an additional 300 automated test cases to cover new functionality

Case Study #1 Results

“LogiGear has delivered an innovative turnkey solution that is fully automated, low cost and has expanded our testing coverage by 90%.

“While reviewing any test automation initiative it is important to understand that a successful automation project requires care and maintenance. Not only does LogiGear provide MX Logic with an excellent automation tool in TestArchitect, it provides me with the cost effective resources to enhance and maintain our automation projects. With a comparable feature set to more expensive licensed tools, my automation budget can be utilized on both the tool and the engineers to support it for the same price. And this guarantees the success of our automation, where others fail!”

- Jamie Tischart, Director,
 Quality Assurance, MX Logic

Coverage expanded from 50% to 90%

Automated test coverage expanded from 10% to 80%

Twice as many tests tested in less time (150 machine hours vs. 240 person hours)

Added regular build-acceptance testing

Added production portal performance testing

Expanded testing coverage and effectiveness while avoiding significant additional expense.

that MX Logic will be adding to their software. This will bring the total number of test cases to nearly four times the company's original manual test suite.

Software Testing 3.0 Case Study #2

Testing Mature Software that was Never Tested

Environment Prior to LogiGear:

A networking software company was testing core back-end functionality of their product manually utilizing a high-cost, on-shore based testing partner using manual testing methods. Because of the time consuming nature of manual testing, and the high cost of the on-shore manual testing partner:

- The company was not testing the GUI component of their solution that they deemed to be “mature” and not subject to a lot of change.
- The company's customers were finding GUI bugs with every new release.

LogiGear's Solution

LogiGear implemented an automated testing program that made use of:

- Low-cost off-shore LogiGear testing resources in Vietnam. These resources are all trained by LogiGear University in Software Testing 3.0 concepts and tools.
- TestArchitect™, a tool set that integrates the latest methodologies and technologies in one easy-to-use package.

LogiGear has created an automated software testing program for this customer that:

- Has implemented 4,000 automated software test cases for the GUI software
- Expanded testing coverage from 0% to over 90% of the functionality of their application's GUI
- Runs the entire automated test suite in only 20 machine hours
- Finds bugs that were previously found by the company's customers helping to improve customer satisfaction with the software and lowering the burden on the vendor's support staff

Case Study #2 Results

4,000 automated test cases

Coverage expanded from 0% to 90%

Automated GUI testing in only
20 machine hours

Case Study #2 Next Steps:

LogiGear's testing resources are now moving on to automating testing for search and data import functionality. In the future, LogiGear will also be implementing an automated software testing program with low-cost Vietnam-based testing resources for the company's back-end software replacing the existing expensive on-shore manual testing.

Generally accepted industry estimates indicate that bugs found by customers are at least 10 times more expensive than those found by software testing prior to release!

For more information, see this article:
Quality Doesn't Just Happen,
Judy McKay, CIO, May 24, 2007

Conclusion

Software testing has undergone an evolutionary process. The early phases emphasized tactics and tools and yielded sub-optimal results. As with any evolutionary process, it is imperative for an organization to continue to evolve, or perish. Software Testing 3.0 is the next phase in the evolutionary development of software testing.

Organizations that make the transition to viewing testing as a strategic initiative — organizations that put testing on a even footing with development by making testing its own organization with its own budget, place methodology and strategy first and then move on to tool selection — will be able to turn quality into a competitive advantage. They will have happier customers. They will have a happy sales force with less turnover. They will meet or exceed their marketing and promotional claims. They will deliver quality software to channels and customers in a timely manner. They will minimize potentially damaging product failure.

As demonstrated in the case studies, rising to the next level allows companies to test faster and better while lowering the costs of testing, support and development. Now being widely adopted in the industry, Software Testing 3.0 is an evolutionary imperative for any software organization.

Source Notes

1. Worldwide and U.S. Offshore IT Services 2006-2010, 2006, IDC
2. Worldwide Distributed Automated Software Quality Tools Forecast and Analysis, 2003-2007, 2003, IDC
3. Global Software Test Automation, 2006, Nguyen et. al, Happy About
4. The Economic Impacts of Inadequate Infrastructure for Software Testing, 2002, National Institute of Standards and Technology

LogiGear Contact Information

For more information, please see at www.logigear.com or call +1 650-572-1400
LogiGear Corporation, 551 Pilgrim Drive, Suite A-1, Foster City, CA 94404 USA
Copyright © 2008 LogiGear Corporation. All Rights Reserved.