

To Use The Silver Bullet... You'll Need a New Gun!

*An exclusive
white paper by
SophLogic
consultant Lee
Stewart*

Executive Summary

During the last 60 years or more, computer technology has evolved exponentially. Computer vendors continually bring to the marketplace more and more complex computational systems that provide more capabilities than anyone previously considered possible.

Software vendors typically develop and market what the computer industry loosely refers to as a *Silver Bullet* solution, implying that their new tool will provide immediate relief to a business process issue or technology challenge. All too frequently organizations purchase these Silver Bullets, only to later learn there is a significant amount of homework/groundwork that needs to be performed before the promised benefits of the Silver Bullet can be realized.

Many organizations struggle - if not fail all together - in their attempt to adopt and implement their chosen Silver Bullet, because they purchased it without realizing that in order to bring to bear all of the benefits the Silver Bullet offers, they may first need to build a new gun and learn how to use it.

The Testing Industry Status Quo (Manual and Automated)

With the growth in computer system capabilities and complexities, the industry began to realize the need to test new or revised systems to insure that they functioned properly. Testing began to be recognized as a necessary process in the more commonly used and industry recognized *Process Models*. Over time, the Process Models began to define the testing process.

Manual Testing

Cultures and organizations naturally resist change to an existing environment. Comfortable in their surroundings, any attempt to change the status quo is considered threatening. Frequently, the challenge of overcoming cultural resistance is the most formidable obstacle preceding successful implementation of a manual testing process or improvements.

TABLE OF CONTENTS

[The Testing Industry Status Quo \(Manual and Automated\)](#) 1

[The Silver Bullet Misconception](#) 5

[What Caliber is your Testing Process? Or, Do You Have One?](#) 6

[Does the New Silver Bullet Fit your Current Testing Process?](#) 7

[Manage Testing Using the Silver Bullet](#) 8

[Build the New Testing Process with the Silver Bullet!](#) 9

[Don't Forget The Rest Of The Testing Kit!](#) 10

[Train 'Em Well - or Feel the Pain!](#) 13

[It Costs More to Turn Your Old Testing Process into an Automatic One - Or Does it?](#) 13

[ROI from the New Process, Training, Testing Kit and the Silver Bullet!](#) 15

[Leverage Experienced Assistance](#) 18

Many organizations attempt to implement testing in a fairly informal, manual fashion that yields only minimal success in finding issues in the product undergoing test.

Failure of the product - in the field or production - soon brings attention to the fact that the testing process is failing to find the problems. This lack of process leads to application issues that are only revealed after so-called successful completion of the test.

In most organizations, when this occurs, someone is assigned to research the *Industry best practices* to determine what can be done to improve the testing process. Perhaps a good process model or two is discovered at this point. But the organization also learns of the complexities inherent in any industry-touted testing process. All too often, the organization then tries to **implement some of the process**, hoping this will appease those who ordered the testing while still resisting changes to the processes. But because they find themselves baulking at implementation of parts of the process, or worse yet, preventing the implementation of the majority of the proven Industry Best Practice for successful testing, **the entire testing project is doomed to fail.**

Industry best practices defining manual testing process approaches can come from a variety of sources:

In the **government sector**, contractual requirements normally identify standards or process models to be followed by contractors:

- 1) The Department of Defense founded some of the more serious process improvement initiatives in the late 1970's, resulting in the establishment of the Software Engineering Institute and the Capability Maturity Models. If your organization is doing business with the DoD, your contract probably contains Software Quality Assurance (SQA) and Testing requirements stemming from documents such as "DoD Directive 5000.1", DoD Defense Acquisition Guidebook, and CMMI, etc.
- 2) The Federal Aviation Administration has testing requirements for organizations building airplanes or subsystems and components for them.
- 3) The Federal Drug Administration has testing regulations for organizations producing products marketed in the medical industry.
- 4) Recently, even the Office of the President of the United States has published "OMB Memorandum M-05-23", which affects all CIOs throughout all federal agencies, their IT/IS organizations and their contractors.

In **commercial industries**, testing process requirements emanate from a variety of sources and in some cases may apply simultaneously:

- 1) The Securities and Exchange Commission sought a lasting solution from Congress targeted at protecting investors from a recurrence of the Enron stock market scandal resulting in the Sarbanes-Oxley Act of 2002, Section 404. This is considered some of the most far-reaching legislation ever enacted by Congress as it affects all publicly traded companies regardless if they are headquartered within the USA or abroad.
- 2) The European community is not shy about requiring companies to register as compliant with ISO Industry Standards (ISO 9000/9001,

Failure of the product - in the field or production - soon brings attention to the fact that the testing process is failing to find the problems.

12207, etc.) Those who are not registered run the risk of other organizations refusing to do business with them.

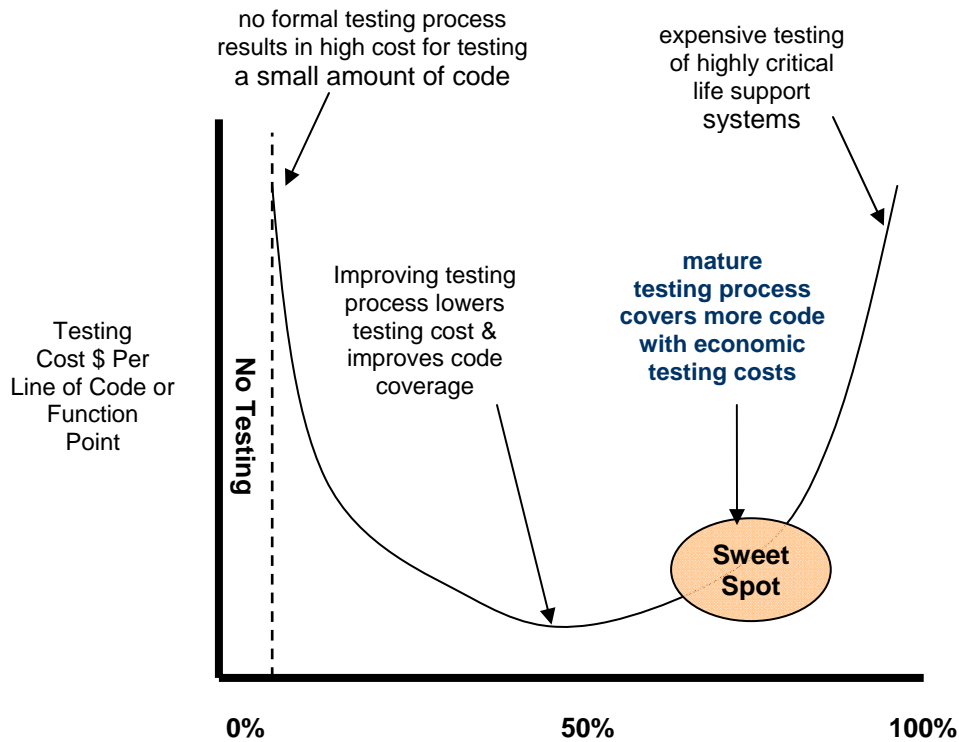
Finding the Manual Testing Sweet Spot

As organizations evolve their testing processes and gain support from top-level management, they eventually reach the point where the majority of issues in a product can be identified and solved prior to shipment.

But finding that sweet spot where costs and testing effectiveness are balanced can be more difficult than originally thought.

The testing industry uses code coverage metrics to determine the amount of code that is actually tested versus untested. The chart below indicates the achievement of improved code coverage as a manual testing process improves (from left to right). **But notice that the cost of testing also increases significantly as the amount of code covered in manual testing increases.**

Code Coverage Achieved with Improvement in Manual Testing Process



Higher Code Coverage in Testing Provides Reduced Risks to Users

As the manual testing process improves and higher levels of code coverage are achieved in test case design, the quality of the product improves. Users are pleased that the application works well right from delivery. But the costs and resources

required to perform good manual testing grow and grow to the point where someone inevitably asks:

“Is there a software tool that can automate our testing - and get it done at a lower cost?”

The answer is: Yes - if implemented well and used efficiently.

Automated Testing

As industry best practice process models containing testing concepts matured over time, automated testing also matured and gained mention in the process models.

Is there an industry standard for automated testing today? Let's just say the jury is still out. But they are getting closer.

To put the concept of automated testing in perspective, the computer industry itself has only been around for 70 years or so. Software quality assurance and manual software testing began to be seriously discussed in the early 1970's. Automated testing processes and related industry standards are considerably younger than either of these and still fairly new.

Despite its short history, automated testing *can* be implemented effectively and efficiently. However, cultural resistance must be overcome for automated testing and associated process improvements to be put in place and make good automated testing a reality. **Having a manual testing process already in place is a good starting point. But automated testing and the processes for effective use are different from manual testing. Period.**

Without adequate management backing, the project may lack process and a formally chartered project plan for implementation, or may suffer from insufficient funds for training...

Strong management backing is key to the implementation of any type of process - manual and automated testing included - so be sure you have the backing of management in writing before you start. With sound planning and budgets for the implementation of automated testing, success can be achieved. **Chartering the implementation of automated testing with a *formal project plan* can begin to lay the appropriate groundwork and carry a strong message from the organization's management team** to the other players. Automated testing must be viewed as an important goal for management.

Without adequate management backing, the project may lack process and a formally chartered project plan for implementation, or may suffer from insufficient funds for training in the use, care and feeding of the automated test tools. An organization's halfhearted attempts to leverage automated testing will fail, and the tools end up abandoned and assigned blame for the failure.

The remainder of this paper is dedicated to serving as a guide toward successfully planning, funding and achieving the implementation of automated test tools and their processes. With this paper as a guide, you'll be better prepared to recognize testing land mines before you step on them, helping your organization avoid the pitfalls of implementing a successful automated testing processes.

The Silver Bullet Misconception

Commercial software development firms began to develop and market so-called automated test tools over two decades ago. And while certainly the test tool industry and the tools themselves have improved dramatically over recent years – here's what you need to keep in mind – they do not *automatically* test any system contained in the inventory of computer systems owned by any organization or business

When an organization starts to search for an automated test tool, vendors will eagerly line up at the door bearing presentations and demo copies of their tools. They'll tell you that they – and only they – possess the Silver Bullet for solving your testing dilemma. **“Let the buyer beware!” should be the first thought in your search for a test automation solution.**

Selecting a Vendor

To evaluate vendors effectively, you must be armed with knowledge about your organization's testing needs and the right questions to ask providers:

- Some automated test tools are built for use with specific operating systems or programming languages. You need to know which O/S and programming languages are used in your organization and which of these platforms contain applications you will be frequently testing – taking into consideration both present and predicted future needs.
- Additionally, some test tools are made to work with particular commercial off-the-shelf (COTS) software packages such as SAP, Peoplesoft, and Oracle. Is the vendor offering a solution that addresses the type of software applications (custom vs. COTS) you need?
- Does the test tool vendor offer integrated automated and manual testing process solutions that can be used for both custom and COTS or have options for variations in testing processes that are adaptable for both?
- Can the tool help you improve organization and management of your manual testing efforts as well? You may have to transition to automated testing using a Proof of Concept approach.
- Some test tools have test data generation capabilities. But none of them automatically produce this test data. Someone must know how to use the tool in order to produce valid data that can then be fed to the test script contained in the automated test tool.
- Can existing manual testers automatically use the new Silver Bullet testing tool? Additional skill sets will need to be acquired by any existing testers before they or anyone else can successfully use automated testing tools.

They'll tell you that they – and only they – possess the Silver Bullet for solving your testing dilemma. “Let the buyer beware!”

At this point ask yourself...does anything mentioned here sound *automatic* ?

What Caliber is your Testing Process? Or, Do You Have One?

The automated test tool brings with it opportunities for your organization. But they may appear more like dilemmas to some people. Some of the questions that should be asked before you obtain the automated test tool are related directly to your organization's current manual testing process.

A. There are several aspects of a successful manual testing process that support the establishment of automated testing. Without these in place, you're going to face an even more difficult mountain to climb. Ask yourself:

- Do we document test plans for each project, system and release?
- Do we document testing requirements and test specifications as a deliverable?
- Do we have a readily reusable testing infrastructure platform in place?
- Do we have a testing organization dedicated to performing testing?
- Are test phases routinely included in all program and project plans with adequate resources and time allocated for testing?
- Are testers routinely trained in improved testing methods and techniques?
- Have all developers and managers been trained in the existing testing process and the reasons for doing it?
- Do releases or bug fixes routinely bypass formal testing?
- Do user organizations constantly complain about system outages in spite of the testing that is performed?

B. If your organization is suffering from the following symptoms, you may need assistance with your manual testing process and other process areas, before considering the use of a automated test tool or attempting the implementation of test automation:

- ➔ A difficult defect that was fixed at great expense suddenly reappears.
- ➔ The system worked in production yesterday, but not today.
- ➔ A fully tested program suddenly doesn't work.
- ➔ A developed and tested feature is mysteriously missing.
- ➔ No one knows which modules/programs/components comprise the delivered system.

The symptoms outlined in Section B are indicative of deficiencies in processes outside the usual realm of a testing process. These symptoms indicate a lack of adequate configuration management, code migration, version controls, quality assurance, architectural design, infrastructure integration, or other such factors.

- ➔ Developers are working on the wrong version of code.
- ➔ The new infrastructure platform is suddenly/unexpectedly too small to support production.
- ➔ The latest version of the code can't be located.
- ➔ There is no traceability between the requirements, documentation, code and infrastructure.
- ➔ The system was built and tested with the wrong version of code on more than one occasion.

Does the New Silver Bullet Fit Your Current Testing Process?

On too many occasions, organizations spend a lot of time *evaluating* the automated test tools and a lot of money *buying* the automated test tools, while very little money is invested in *preparing* the organization for the automated testing process. Forgotten steps also include providing an adequate platform for using an automated test tool and training for a smooth transition from the existing testing status quo to an efficient automated testing process.

If you don't have a formalized manual testing process in place, your organization probably doesn't have anyone on board who has the skills or knowledge base to design a testing architecture for any project's testing. Your organization may not have anyone who understands how to design a test case or test data set for it for either manual or automated testing.

If test design concepts are not a well understood subject or an existing skill set within your organization, buying an automated test tool will likely become a very expensive way to prove that nobody in your organization knows anything about testing. But don't be discouraged. Your organization can still experience the benefits of automated testing by drawing on the expertise of a knowledgeable third party for assistance until your internal resources reach the point where you can effectively manage the testing tool without external assistance.

Forgotten steps also include providing an adequate platform for using an automated test tool and training for a smooth transition from the existing testing status quo to an efficient automated testing process.

Additional cultural resistance to the implementation of testing is especially foreboding if you have no existing formalized manual testing process in place. **Some organizations unfortunately attempt to kill two birds with one testing Silver Bullet - they buy an Automated Test Tool with the hopes that it will cure both their lack of a testing process and automatically allow them to perform efficient testing.** This misconception has all too frequently set up many an organization for disappointment.

The automated test tool and the selection of it usually reveals additional associated dilemmas beginning with the first vendor test tool presentation or test tool demo... Provided you've previously studied industry best practices defining testing concepts and/or perhaps understand the details of manual testing processes in your

organization's policies, you may suddenly notice this problem while you are listening to the test tool vendor's presentation:

The test tool vendor representative may seem to be speaking in a "foreign" testing process language. Indeed, most of the tool vendors have their own testing process lingo, an automated testing language, and it usually comes with their testing process built in by default!

Consider this...does the automated test tool force drastic changes to your way of talking about testing? Or does the automated test tool facilitate adopting the test tool to your testing mentality or vocabulary...some test tools are better at this than others.

Manage Testing using the Silver Bullet

Today's matured test tool vendors have realized the need for an automated test tool and its related testing process to be fully integrated with other processes and teams within an organization.

Suites of automated test tools are now being developed and published containing capabilities intended to be very useful not only for testers but for other players in the organization as well, including testing managers, project managers, software development managers, development team members and senior management.

These suites of tools include testing management capabilities. Scheduling tools are being included that allow **work assignments** to be made to any tester tasked with a specific test case/script development and/or execution.

Defect or bug tracking capabilities are included in most tool suites as well. Here the tools allow testers to enter a defect/bug and track the status of each of these to resolution. Not only should this functionality simply track each defect/bug, but it should also communicate (via email notification functionality) to other members of the project team the existence of the defect/bug. Task assignments can then be made by the appropriate management team members to the specific developer assigned to resolve the coding or design issue indicated by the defect/bug.

These testing management and defect tracking tools usually provide reports of statistical information regarding the progress being made in testing for each project/system undergoing testing, for example:

- ▶ How many tests are scheduled to be developed?
- ▶ Which tester is assigned to develop or run each test?
- ▶ In which sequence are the tests to be executed?
- ▶ Has the test data been prepared for each test?
- ▶ Has the test environment been setup for each test?
- ▶ How many tests are complete vs. incomplete?
- ▶ How many defects have been found?
- ▶ Who is assigned to resolve each defect?
- ▶ How many resolved defects are ready for retesting?
- ▶ Who is assigned to retest each resolved defect?
- ▶ How many resolved defects have been successfully retested?

Don't be among the organizations that initially invest in only the automated testing scripting tool and overlook these test management and defect-tracking test tool suite options.

- ▶ How many defects have been resolved vs. retested?
- ▶ Which defects are high risk or high priority for the pending release?
- ▶ Which tester has too many tests assigned to him or her?
- ▶ Which tester is available for new testing assignments?

Don't be among the many organizations that initially invest in only the automated testing scripting tool and overlook these test management and defect-tracking test tool suite options. The industry has learned that defects cannot be effectively assigned and successfully resolved via regular email, as it consumes too much time and effort to track testing status and defect status via a bundle of regular emailed assignments, log sheets or spreadsheets for the status of each project/system or other detached and clumsy methods.

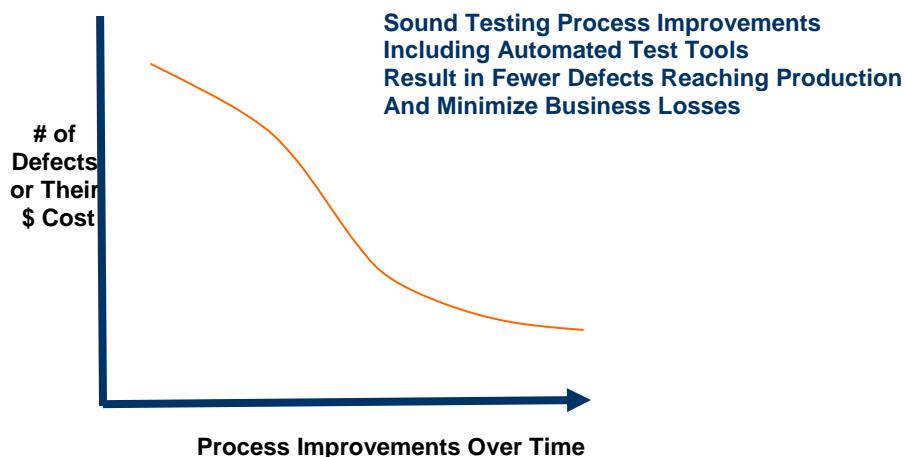
Build the New Testing Process with the Silver Bullet!

Higher quality test tool suites contain capabilities that facilitate successful adaptation of the test tool suite to the organization and its culture.

Although many of these tool suites come with a predefined testing process by default, the better test tool suites enable your organization to modify or enhance the default testing processes contained in these tool sets. This is very valuable in assisting the implementers in overcoming cultural resistance to using the tools while improving an organization's testing process.

The risk of cultural resistance can be reduced when functionality like this is included in the organization's test tool suite selection criteria. A more robust test tool suite can help design a more culturally acceptable test management, test development, test execution and defect resolution process.

When improving manual and automated testing processes, the testing process should be linked to other process areas, including project management, configuration management, and development management at a minimum. Improving the testing process in isolation from these other process areas will not help the organization or the business to the extent it should. Modern automated test tool suites can readily be linked to other process oriented tools via APIs and the like.



During the last few decades, the computer industry has learned that multiple processes need to be adopted, improved and integrated with each other if we're to deliver systems on schedule, within budget and working correctly.

If systems are delivered to production with problems, the business area can't do their job function well, if at all. The more problems we have in the production environment, the more business losses are incurred.

With improved testing processes, these problems can be identified at a point in time where they can be fixed before they create an adverse effect on the business, business area users, management and the bottom line.

Don't Forget The Rest Of The Testing Kit!

During the selection and implementation of an automated test tool suite and while working toward improvements in the testing process, testing organizations should remember **there may already be additional systems/platforms used by other managers in the organization** as they plan, fund, develop and track programs, projects and related activities necessary to facilitate management of enterprise level development, maintenance, operations and production support activities.

An automated test tool/suite is a very exciting prospect for testing organizations. But in that excitement, they may overlook or forget about the need to integrate testing process improvements and the test tools with other pre-existing technology and management processes outside of testing.

Enterprise-Level Management Needs

Senior managers in the organization are challenged with keeping track of budgets and planning for all enterprise-level activities. It's important to examine how the testing organization and proposed improvements to the testing process should be integrated with these pre-existing management processes, the capabilities used to support them and any expectations associated with continued use/support of these or participation in them by the testing organization and testing managers.

This organization-level management integration with testing management should begin while planning the test tool implementation project, continue while collecting requirements for the test tool selection, be considered as an important factor during the test tool selection process, designed into proposed improvements in the testing process, configured into the test tools while implementing the test tools of choice and exercised during all follow-on test planning, test management and testing budget determination activities.

Test managers and testing processes should be integrated with other managers and processes such as:

- Project Management
- Software Development Management
- Configuration Management
- Security Management
- Infrastructure Management

- Operations Support Management
- User Representatives
- Business Area Management
- Senior Management

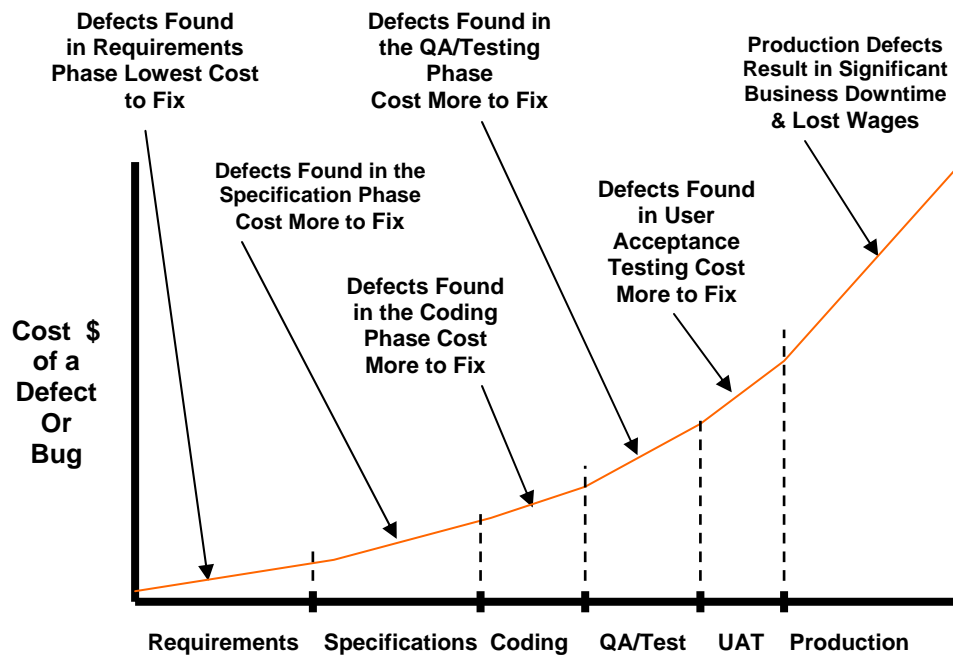
After all, testing is being performed to determine if all requirements have been met in the product being tested. All of the managers listed above have contributed to that set of requirements or have a vested interest in them.

Full Spectrum Quality Assurance Process Needs

With the evolution of testing in some organizations, the testing group became labeled the QA or Quality Assurance group. But do they really do QA?

In the more established process models there are methods and activities referred to as Quality Assurance, QA or perhaps Verification and Validation. These methods extend beyond the usual boundaries of testing. By becoming skilled in these techniques and implementing them, a true full spectrum QA group can assist its parent organization in achieving even more significant reductions in costs associated with the development, test, maintenance and support of systems throughout their lifecycles.

Defects in a typical testing group usually only refer to issues discovered in test phases. But by using true QA processes in conjunction with improved testing processes, defects can be discovered earlier in the Software Development Lifecycle and resolved at a much lower cost as depicted in the Cost \$ of a Defect chart below.



Typical Software Development Lifecycle Phases

Evolving the testing group toward becoming a full spectrum QA group can be more easily accomplished with the use of automated test tools and by testers becoming familiar with the usual CASE tools employed by the developers and architects for design and coding. True Quality Assurance methods usually employ techniques such as Peer Reviews of Deliverables, Code Walkthroughs, Design Reviews, etc., all of which allow the QA group to identify defects in the earlier phases of each project. Additional

benefits follow while the QA group is participating in these activities, as the testers will learn early on about the systems functionality they will be testing and allow them to prepare for the tests in a more efficient manor.

Testing Infrastructure, Platforms and Environment Needs

In order to efficiently perform manual or automated testing, **a dedicated testing environment, test lab or test facility needs to be permanently established.**

This facility and environment should consist of an adequate numbers of servers, workstations, and associated peripheral equipment.

The test environment design and requirements should reflect the production platforms to some extent. An adequate inventory of dedicated test equipment should be readily available, so that the test environment can be re-configured to facilitate testing of various projects/systems as they are being prepared for migration to production platforms.

As automated test tools are brought into play, be sure to have a dedicated platform for them to reside on. Test cases and test scripts are highly reusable commodities. As automated test scripts are developed, the library or inventory of highly re-useable automated testing assets will grow and grow. As automated test cases and test scripts are developed and used for each project coming down the pike, these need to be retained and protected for future reuse opportunities. An adequate platform, archives and back-up capabilities need to be established to facilitate ready access to these re-usable assets.

Additional testing environment requirements need to be addressed as well, including:

- Software tools used to build the testing environment
- Code migration tools and processes
- Configuration management tools and processes
- Software build tools, loaders, linkers, etc.
- Version control tools
- CD WORM drives for backups and restores
- Printers representative of the production printers
- Routers, switches, gateways, etc., representative of the production environments
- Connectivity between the test lab platforms and the production environment to facilitate copying production databases into the testing environment when needed for testing, etc.

It's important to include hardware in the test lab that represents "the good, the bad and the ugly" aspects of the current production environment/infrastructure. If we only test proposed systems on the fastest hardware available, testing in this manner may not verify whether the new system is capable of running on the older/slower platforms that still reside in the targeted production realm.

It's a good idea to have a representative sampling of the various production platforms available in the test lab. This is especially true when using automated performance/load/stress testing tools in the test lab in an effort to determine whether the current/proposed production platforms will support the software systems that have been proposed to run on those selected production platforms.

In order to efficiently perform manual or automated testing, a dedicated testing environment, test lab or test facility needs to be permanently established.

Train 'Em Well - or Feel the Pain!

If you were buying a \$1 million technically complicated Enzo Ferrari, would you hire a Shade Tree Mechanic to work on it for you? Hopefully, not!

In the complex world of the computer industry today, why is it that some organizations buy new complex tools or technology and then toss it at the employees, saying "Just read the User's Manual...You'll figure it out!" Does the phrase "failure to plan means planning to fail" spring to mind?

In today's complex computer industry testing processes, automated test tools and test suites have become complex high technology and require technical skill sets for their successful use. **Gone are the days of old where you could just get together a group of users and task them to test a system with vague instructions like, "See if you can break it...Pound away at the keyboard...Just tell us if something breaks!"**

If you're going to improve processes and implement new tools to support the new process, be sure to budget adequately for training and train everyone who will be installing, configuring, maintaining and using the tool hands-on, as well as those responsible for managing any resources associated with the tools or processes.

When organizations are faced with the adoption of newly purchased automated test tools, they should immediately realize they are going to be technologically challenged. Even the existing testers may have become very comfortable while working under the old testing process and now they are concerned about the loss of the status quo. That's right - even the testers may exhibit some cultural resistance toward the changes in the testing process and the new test tools.

What about the developers in the organization? They are now expected to resolve defects/bugs that are being assigned to them and tracked to resolution in the new automated test tool suite they've never seen before! While being told that the Software Development Manager now will have visibility of the existence of all the defects by being able to readily obtain a list of all defects and find out or assign the responsible developer to resolve each of them. Imagine the culture shock on the part of the developers!

If you're going to improve processes and implement new tools to support the new process, be sure to budget adequately for training.

It Costs More to Turn Your Old Testing Process into an Automatic One - Or Does it?

Until the cost-saving benefits of sound testing processes are understood, the sticker price of automated test tool suites organizations can sound high. The high entry level costs associated with the acquisition of automated test tool suites and the skilled testers necessary to use them surprise many organizations – **but the positive effects can far outweigh the costs.**

Seldom do organizations realize the costs associated with not testing well. The immediate concern is that they will spend too much money on testing, as though it would be wasteful of their technology budget dollars. **The fact that they may be wasting much more money by not testing properly often escapes executives.**

Without training on the cost-benefits of testing, organizations usually won't understand the associated cost of *not* testing well. Not testing or poorly orchestrated testing is even more expensive to the business as a whole than spending some money to improve the old testing process, acquiring the automated test tools, implementing the new testing process and tools and then performing repetitive test cycles for each project or release of each system.

It's All About the Money!

When an organization has implemented enterprise levels of high tech, costly and complex computer systems, they are often faced with tight project timelines and increased demands from the business to provide additional computational capabilities – and to do so on a limited budget.

Too often technology organizations will incorrectly conclude that the cost of fixing a production problem would be no more costly than finding and fixing the problem with a good testing process. If they only understood the overall cost to the business, they would see this could not be further from the truth.

A production problem impacts more than the IT department - it impacts the business. When a system goes down in production, this leaves the business users in a position of not being able to perform their business function. During an outage, the business is paying wages to the users who may be able to do little more than sit around twiddling their thumbs.

Consider this:

When a system fails to perform properly in the production environment, the organization responds quickly to address the business's need for an immediate solution.

Resources are reassigned quickly to respond to the problem. The organization finds itself working in fire-fighting mode, sending resources to work on the issue appropriate for an emergency response.

As a result, projects that were being worked on in accordance to their project plans are abandoned in favor of the rescue effort.

Resources assigned to putting out the fire are now unavailable to participate in their planned activities where they were working before the crisis arose. **Tasks are forgotten, meetings are delayed, deadlines are missed, project timelines are extended, and project budgets run over** as other team members await the return of the missing players.

Calculating the cost of downtime

When you research the number of times per year that a given system goes down and the duration of the downtime per occurrence you can calculate the total hours of downtime endured by the affected business areas annually.

Using the sum of the total hours that the system is down per year multiplied by the number of business users that were put out of action, the resulting calculation is then multiplied again by the average fully loaded hourly cost per business user (including their wages and benefits package). The result represents the cost of the downtime in terms of lost wages alone.

Additionally, when the users are finally able to log back in, a backlog of work now awaits them. Instead of their normal productivity, they are behind on their work. The business managers realize that the backlog needs to be caught up. The number of users hired was based upon an average amount of work to be accomplished each day, while assuming the system would be available. Hours of overtime (often paid at time and a half) will be paid to catch up on the backlog. Business users have now been paid straight time for not doing anything, plus time and a half for catching up on what they would otherwise been able to accomplish.

...So the outage has now cost the business two and a half times the usual fully loaded wage and benefits cost.

But the cost to the business as a whole doesn't end there. While the business users were unable to perform their job function, what else was impacted? Perhaps the affected business is responsible for marketing or sales. Without the availability of their core systems, there may have been missed opportunities or lost sales. While not as easy to quantify in hard dollars, these losses impact business revenue.

There are potential domino effects of the recurring system outages on an annual basis which impact the business as a whole:

- While the system was down, were the other systems - that usually receive data from it across the systems interfaces - unable to obtain current or accurate data across the inoperative interfaces?
- Were business decisions made during the downtime periods that may have been based upon inaccurate or, worse yet, corrupted data?
- If poor business decisions were made, did they have an ill affect on the cash position of the business?
- Were orders placed to increase inventories of the wrong goods?
- Were decisions made that created an ill affect on the bottom line?

It doesn't take many occurrences of these issues to fully offset the costs associated with improving testing processes and purchasing automated test tools.

It doesn't take many occurrences of these issues to fully offset the costs associated with improving testing processes and purchasing automated test tools.

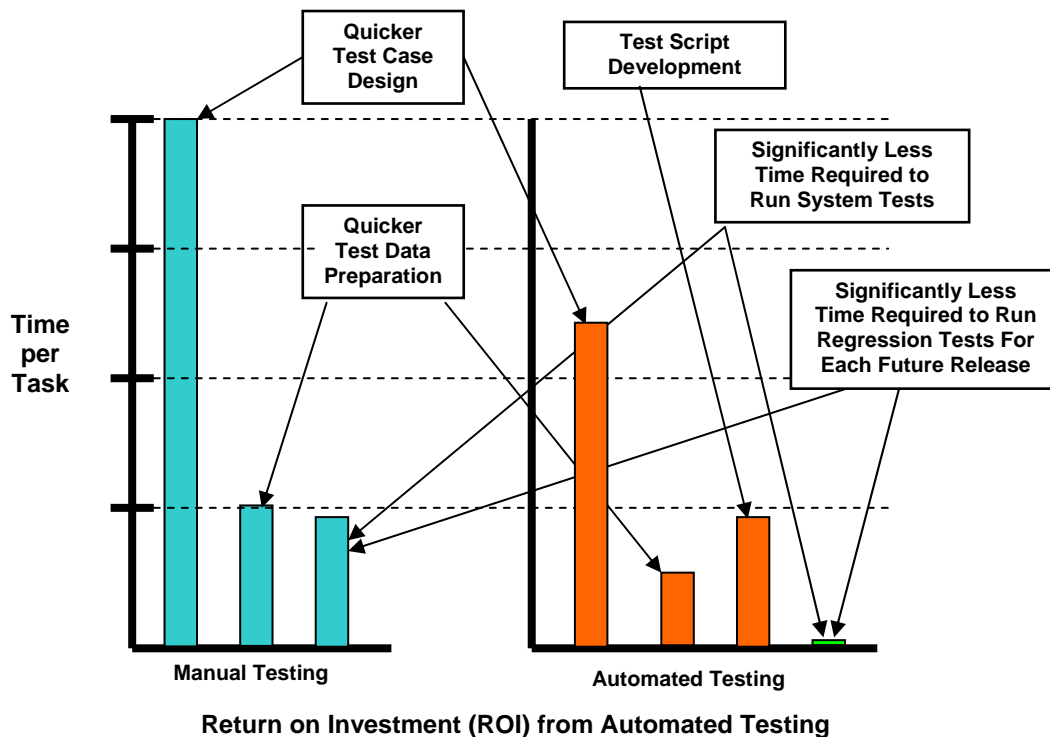
ROI from the New Process, Training, Testing Kit and the Silver Bullet!

With sound manual and automated testing processes, the right automated test tools, skilled testers, a robust test tool kit and an adequate testing infrastructure platform in place, software defects can be reduced.

You should know

- Finding and fixing a software problem after delivery is often **100 times more expensive** than finding and fixing it during the requirements and design phase.
- Software projects spend about **40 to 50 percent of their effort on avoidable rework**.
- About **80 percent of avoidable rework** comes from **20 percent of the defects**.
- About **80 percent of the defects** come from about **20 percent of the modules** of code and about **half of the modules of code are defect free**.
- About **90 percent of production down-time** comes from about **10 percent of the defects**.
- Peer reviews catch about 60 percent of defects and at a much lower cost.
- Disciplined personal practices can reduce defect introduction rates by up to 75 percent.
- About 40 to 50 percent of user programs contain **nontrivial defects**.

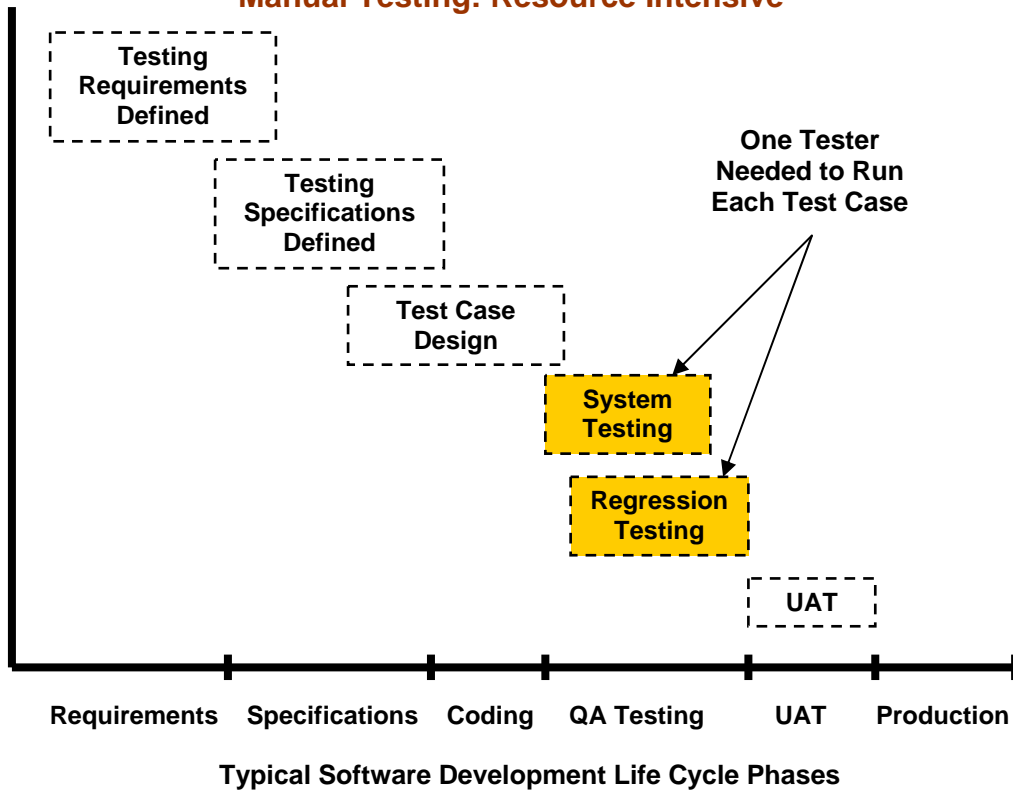
The costs associated with testing itself can be significantly reduced through the implementation of automated test tools. The following figures compare the costs and efforts associated with manual vs. automated testing methods.



Manual testing methods can be employed to significantly reduce risks to the production environment and to the business as a whole.

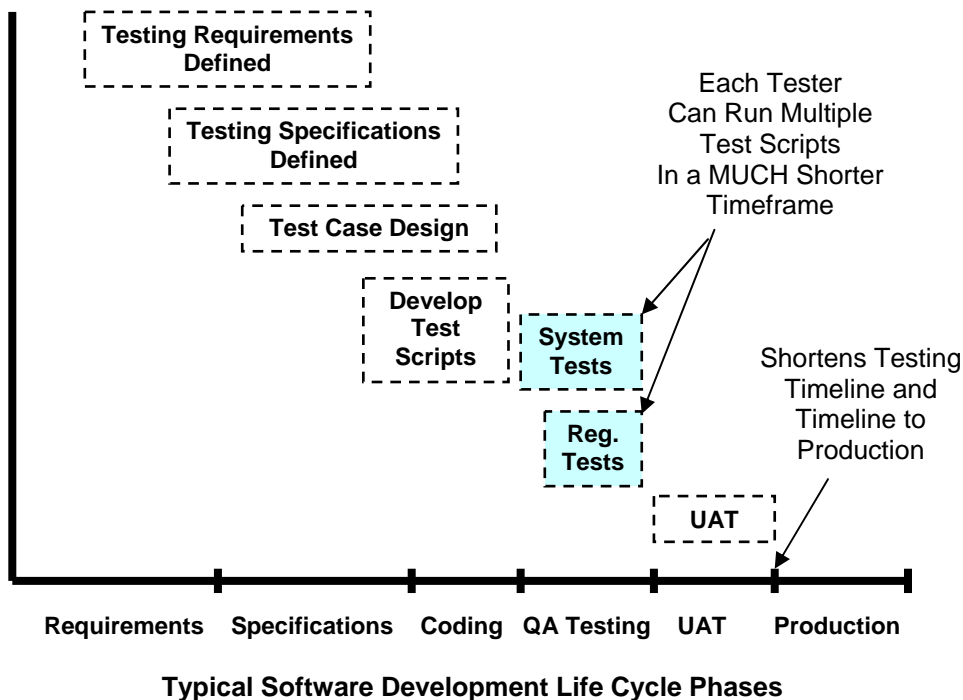
Manual testing requires a single tester be available to run each manual test case. The more test cases required to be executed in a fixed testing timeline, the higher the tester headcount needed.

Manual Testing: Resource Intensive



Manual testing cannot achieve the testing resource and project timeline efficiencies obtainable through automated testing.

Automated Testing: Conserves Resources & Timelines



Leverage Experienced Assistance

To implement adequate manual and automated testing, be sure to obtain adequately experienced assistance from outside sources. A knowledgeable third party can guide your organization toward improved methods and processes by:

- ☑ Training your staff in improved manual and automated testing process methods.
- ☑ Facilitating better control over manual and automated testing projects for management.
- ☑ Facilitating improved estimates of manual and automated testing cost and schedule.
- ☑ Assuring increased productivity of your manual and automated testing project teams is obtained.
- ☑ Significantly reducing project risks through improved manual and automated testing.
- ☑ Reducing cost of development, fixes and rework rates with improved manual and automated testing.
- ☑ Improving the reputation of your organization to foster the attraction and retention of good staff.
- ☑ Increasing customer satisfaction.
- ☑ Reducing risks of legal action from unhappy customers.
- ☑ Reducing business losses with improved defect detection and removal as a result of implementing improved manual and automated testing.

With the assistance of experienced testing professionals, you can build your new gun and use the Silver Bullet effectively.

About SophLogic

SophLogic is a leading SAP and IT professional resources and consulting company. We help organizations derive more value from their SAP investments. As a growing leader in software quality assurance, SophLogic ensures your SAP and IT investments produce optimal business outcomes. Experienced Quality Assurance Leads bring greater value to every initiative. Organizations working with SophLogic can benefit from the industry leading solution for testing and quality assurance throughout the entire business application lifecycle, from implementation and upgrades to patching and consolidations.

SophLogic offers a wealth of SAP and IT industry experience and in-depth knowledge in specialized subject matters, with experience spanning more than 50 industries and bridging every continent. Enhance the performance and availability of your business-critical SAP solutions with a partner who is committed to improving quality, performance and scalability for every SAP application, to help you focus your resources where they achieve the greatest results for your organization. Contact us today to learn how SophLogic can deliver the value and results your business needs.

About Consultant Lee Stewart

For over 20 years as a management consultant, Mr. Lee Stewart has helped organizations define, develop, document, train and implement best industry practices, standards, methodologies, process improvements and metrics. His experience includes: project management, engagement management, client management, configuration management, change management, quality assurance management, inspections, verification, validation, automated and manual testing, code migrations and methodologies for combinations of business processes, software and hardware technology programs/projects.

Through his efforts, many organizations have:

- reaped large business cost reductions with significant reductions in operational/production issues,
- achieved improvements in customer satisfaction, established cost containment for projects and
- increased timely project completion for the development, manufacture and maintenance of systems.

His considerable experience is based upon SAP/ASAP, RUP, CMM, CMMI, ISO, DoD Directive 5000.1, MIL-HDBK-61A (SE), and NASA processes for all stages of the System Life Cycle and Software Development Life Cycle (SDLC).

He guides and assists Government, U.S. Military, commercial businesses and their related IS/IT organizations in achieving compliance with COBIT, ITIL, CMM, ISO 9000/9001, OMB Memorandum M-05-23, MIL-HDBK-61A and Sarbanes-Oxley Act Section 404 for Quality Assurance, Testing and Configuration Management.

As a member of the first Space Shuttle Launch Team, Mr. Stewart managed, negotiated, designed, developed, documented and implemented Software Policies and Procedures for the Space Shuttle Program at Kennedy Space Center affecting thirteen NASA contractor companies and three NASA Engineering Directorates in all phases of the Software Development Life Cycle (SDLC).

Courses authored and presented by Mr. Stewart achieved the highest ratings available from Government Agencies and industry for Configuration Management based upon:

- Software Engineering Institute, Capability Maturity Models (SW-CMM and CMMI),
- ISO9000/ISO9001, Office of Management and Budget Memorandum M-05-23,
- Department of Defense Directive 5000.1, MIL-HDBK-61A (SE), Defense Acquisition Guidebook, and
- Sarbanes-Oxley Act Section 404 (SOX) for Configuration Management and Quality Assurance.

Frequently, Mr. Stewart directs and manages assignments in a customer facing role as the key liaison between customers, strategic partners and project teams; during the development, implementation and support of technical solutions or process improvement initiatives targeting compliance with appropriate regulations/standards.