

# Offshore Software Test Automation: A Strategic Approach to Cost and Speed Effectiveness

## Synopsis

The following white paper presents an executive overview of an innovative approach to integrating global resourcing and the latest test automation methodologies and tools. This approach can effectively help any software development organization meet their goals of time-to-market, cost containment, and quality.

To provide the proper background for this discussion, the paper also summarizes the software development process, the software testing process, global resourcing and the automation of software testing based on a structured approach and methodology known as Action Based Testing (ABT).

This paper is based on the more in-depth treatment presented in the book **Global Test Automation: A Discussion of Software Testing for Executives** ([http://www.logigear.com/about\\_us/books/gta.asp](http://www.logigear.com/about_us/books/gta.asp)).

Whitepaper Synopsis .....	1
Introduction.....	1
Executive Overview of the Software Development Process .....	3
How Quality Fits.....	5
Executive Overview of Software Testing.....	6
Executive Overview of Test Automation .....	8
Executive Overview of Offshore Outsourcing .....	9
Choosing a Global Test Automation Partner.....	11

## Introduction

Two industry trends, automating software testing and moving software testing offshore, hold out the promise of providing both cost and time savings. Bringing software to market faster while reducing costs and helping the bottom line are goals that are very high on any software development organization's wish list.

Unfortunately, many of these efforts yield results that fall far short of expectations. The trade and popular press are full of stories of failed offshore efforts. In addition, many test automation efforts not only fail to yield time or cost savings, but in fact result in quite the opposite! A coordinated management effort, with good understanding of the planning required, is necessary for success in both of these areas.

In fact, the strategic integration of the latest test automation methodologies and technologies *with* global resource strategies will not only improve upon both efforts, it

will allow an organization to fully capitalize on the speed and cost saving potential of offshoring and automation.

The competing goals of delivering a quality software product, reducing costs and meeting time to market targets often lead management to take a tactical approach, focusing entirely on one goal or approach while neglecting to consider the impact of their decisions on other parts of the process. They may simply focus on one dimension such as automating testing to improve time to market, or off shoring testing to drive down costs.

Taking a tactical or one dimensional approach can very often lead to undesirable or unforeseen results. Management needs to consider the interplay between quality, cost, and time-to-market goals. They need to take a more strategic approach and develop an overall strategy and methodologies, and then select tools and partners.

An effective testing automation effort requires:

- A full understanding and appreciation of the software development process
- An understanding of software testing as a strategic effort that can provide cost benefits, speed benefits, as well as providing management with critically important visibility into the quality of their software under development
- An understanding that software testing is a discipline that is separate from development that should have its own budget and funding
- An understanding of the importance of a structured test automation approach that is based on a methodology known as Action-Based Testing (ABT), which creates a hierarchical test development model that can improve the quality and speed of testing while reducing the costs of testing
- An appreciation of the strategic importance of global resourcing strategies and best practices to further drive down costs and impact the bottom line

This white paper will:

- Provide an executive overview of the software development process, and the proper fit of the testing and automated testing processes, to help executive management to achieve a better understanding so that they will be able to make more informed and effective decisions on test automation and global resourcing.
- Discuss the strategic integration of the latest test automation methodologies and technologies such as a structured approach that is based on a methodology known as Action-Based Testing (ABT), with global resource strategies to fully capitalize on the speed, cost advantages, and best practices in automation and global sourcing.

Present the importance of selecting an experienced strategic partner such as LogiGear, who can provide an integrated testing solution based on ABT *and* a global resource strategy that will effectively decrease testing time and cut costs while meeting quality goals

## Executive Overview of the Software Development Process

At its most basic, there are three broad categories of efforts in any software product development life cycle. These are:

1. Specification or requirements definition
2. Development
3. Testing

This simple list, and the prevailing view of the process, implies a linear relationship between each of the components – the process moves from specification to development to testing as illustrated in Figure 1.



Figure 1

This is an overly simplistic view of the process of developing software that focuses on the tasks while ignoring the strategic interplay between the three parts of the process. It is, unfortunately, how the process may be implemented in many companies. There are several problems with this task-focused view of the process, including:

- It assumes that the specification delivered will not change
- It assumes the engineering will deliver software to testing on time and that there will not be any changes
- It involves testing far too late in the process

It is essential to understand that each step in the overall product development life cycle is in itself its own discipline and process. Further, each process is substantially enhanced if it is *informed* by, and continuously involved with, the other steps in the process. In reality, the steps in the development life cycle should both overlap each other and provide feedback to each other. For example:

- As product marketing/planning starts to develop a requirements document (hopefully with extensive customer and user input), engineering can start to investigate alternative ways of implementation, develop initial implementation

estimates, identify critical paths, and start to work on translating requirements into a workable engineering plan.

- Engineering can also provide feedback to those developing the requirements on feasibility, risks, and time to implement certain features and functionality that may trigger alterations to the requirements specification.
- Testing involvement at these early stages provides visibility into what will need to be tested so that a test plan can start to be developed and test cases planned.
- Testing can design and automate test cases in advance so they are ready when development starts to deliver them code to be tested.
- Testing can also provide feedback, time estimates, risk assessments, and identify critical areas or provide feedback on the merits of different ways of implementing the requirements. They can even ask development to put hooks into the software that will make testing easier and faster.
- All will be able to more effectively deal with any changes or alterations and there will be fewer project slowing “surprises”.

Figure 2 illustrates the more parallel nature of the process with the invaluable *feedback loops*. The strategic integration of the three disciplines into a more streamlined and cooperative process helps to improve quality, improve delivery time, and reduce costs.

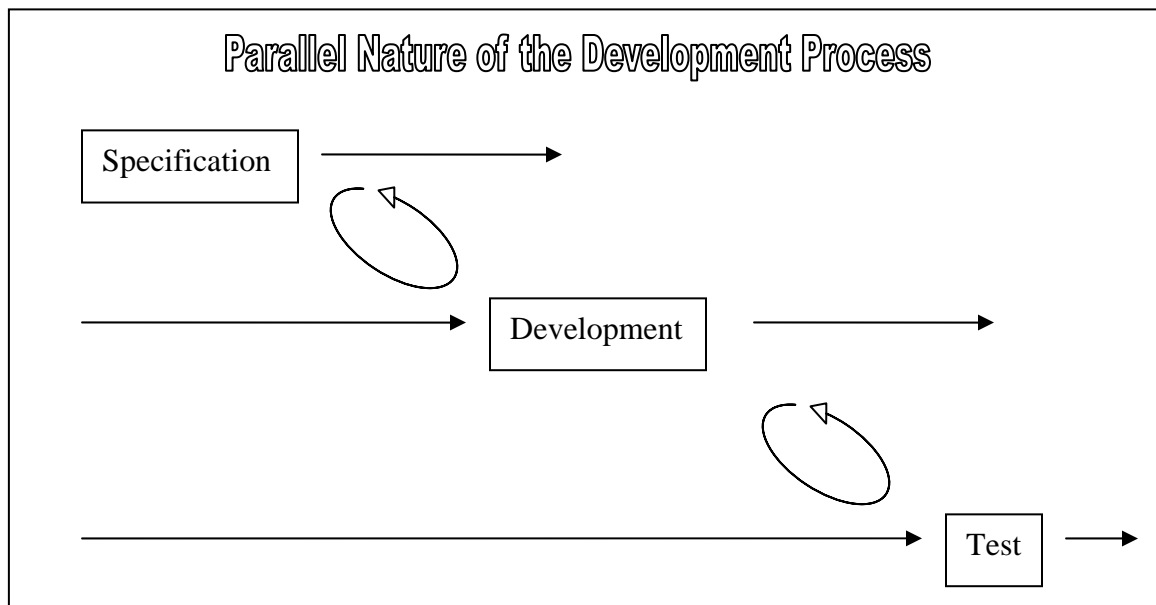


Figure 2

## How Quality Fits

Many incorrectly associate quality assurance with software testing. It is possible to do an excellent job of software testing and still deliver a poor quality product! Poor requirements definitions, or poor implementations, can lead to a product with unwanted or unusable features. Testing may verify that such a product is relatively reliable and works as intended, but nobody wants it or wants to use it. Such a “working” product will be seen to be of poor quality in the eyes of users and the marketplace. Delivering a product like this can have a very negative impact on an organization, driving up costs, and potentially impacting current or future revenue streams.

It is important to understand that quality entails efforts at every step in the development life cycle. It is also important to understand the costs of quality throughout the process, and that costs differ depending on where in the process they are incurred.

*Quality cost*, or the total cost to deliver a quality product, is simply the sum total of all quality related costs incurred at every phase of a project. There are four broad categories of quality costs:

1. **Prevention costs** – Prevention costs represent all costs spent to prevent software, documentation, and other product related errors. These costs include tasks such as staff training, requirements analysis, early prototyping, defensive programming, usability analysis, clarity of specification, and more. *Prevention quality costs are the most cost-effective quality dollars that a company can spend.* In essence, it costs much less to avoid errors up front than it does to identify them later, determine how to fix them, and develop new code to rectify them.
2. **Appraisal costs** – Appraisal costs represent all of the money that is spent on testing activities, which are any and all activities associated with searching for errors in software and associated materials. It includes the testing done by developers themselves, as well as the testing performed by the software testing organization. It is one of the largest costs associated with quality. However, *it is far less expensive to find the errors early and fix the problems prior to software release.*
3. **Internal failure costs** – Internal failure costs represent all of the costs associated with fixing bugs found prior to release. Again, whatever these costs, *it is much less expensive and disruptive to fix problems internally prior to product release than it is after the product is released to customers.*
4. **External failure costs** – External failure costs are all of those costs associated with defects and errors that are discovered after the product is released. These costs include all of the direct costs of identifying and fixing the problems, usually under high-pressure, as well as all of the sales and marketing costs associated with *damage control*. There are also *intangible costs* such as loss of goodwill, low customer satisfaction, and impact on future sales. *External failure costs are very*

*costly*, and are typically much higher than internal failure costs. External failures can also negatively impact an organization's reputation, putting at risk current and future revenue streams. External failures present the very real risk of negatively impacting an organization's bottom line profitability.

The challenge to management is to do a cost-benefit analysis to achieve an optimum balance between prevention and appraisal costs to help to minimize failure costs, reduce overall costs, and positively impact the bottom line.

## **Executive Overview of Software Testing**

Many mistakenly believe that software testing is a small extension of development, and worse, some assume that testing is performed exclusively by software engineers. Others assume that software testing is something that happens at the end of the development process - after the software is developed; it is then tested and shipped. Others equate quality assurance and software testing. All of these beliefs are incorrect.

Software testing is a highly strategic and specialized discipline. It is very different from, but related to, software development (just as sales and marketing are very different but related disciplines). At its most basic, software testing is a concerted attempt to break software so that bugs may be identified and fixed before end-users encounter them. Testing uses skills, methodologies, insights, and creativity that are very different from those used by software developers.

Software testing is a critically important part of the overall development process. Because of its strategic importance software testing should be its own organization, separate and distinct from development, with its own budget. This setup can help to enable:

- Effective training
- Continuous skills and career development
- Efficient implementation of testing tools
- More effective communication from the testing team to development and management

The software testing process has several main activities. These include:

- Designing the software tests
- Running the tests
- Identifying problems and defects
- Reporting to management and development on key metrics

Testing is an iterative process of identifying, fixing, and retesting, as well as reacting to design and code changes. Software testing consists of *both* manual testing and automated testing. While it is true that manual testing should be kept to a minimum; there will always be tasks, such as usability testing, that require manual testing.

**Manual Testing: Productivity Objective**  
No more than 5% of tests should be executed manually.

Software testing is not quality assurance; it is a part of quality assurance. All steps in the development process, from requirements definition, to design and development, share in quality as an objective. The role of software testing in the process is to identify defects so that they may be fixed. You cannot, however, test quality into a poor design (as the saying goes, “bugs may be tested out, but quality must be built in”).

Software testing is also not something that simply happens at the end of the development process. To be effective, software testing needs to be a *strategic partner* with product marketing and development from the beginning of the development process. Software testing needs to have a clear and full understanding of the goals and objectives of the software under design and development. Having such knowledge will help them to design better tests. Early visibility also helps software testing to develop their testing plans, as well as start to design test cases. It helps to decrease testing time and costs.

In addition to being an *activity*, software testing generates *products* that can be viewed as strategic assets to an organization. These products include:

- Test cases that can consolidate the intellectual property of your team members
- Automated tests that can be re-used, becoming assets that reduce costs

Software testing is also a *strategic resource* that should provide a major benefit to senior decision-making executives. Software testing provides visibility into the status of a product under development, its maturity and its readiness for release or production. In essence, software testing is an information service for the executive team allowing them to make *more informed and effective decisions* when it comes to the quality and readiness of their product. Software testing accomplishes this by providing a small number of well chosen metrics to help track testing progress and the quality of the software.

**Testing Metrics Provide Visibility**  
Testing Metrics must provide visibility into a software product’s quality. Metrics are only useful if they help to make sound business decisions. Metrics fall into two categories:

1. Project management
2. Process improvement

Some metrics can measure the output of your test team, such as:

- Bug reports
- Test cases written or test cases executed

Other metrics, such as coverage metrics, give visibility into how much of the software has been tested.

## Executive Overview of Test Automation

Test automation can provide great benefits to the software testing process and improve the quality of the results. The reasons to automate software testing lie in the pitfalls of manual software testing. Manual testing:

- is slow and costly
- does not scale well
- is not consistent and repeatable
- is difficult to manage

While these factors may drive the desire and need to automate testing, it is important to take the *right* approach to test automation. There are several basic steps to automating testing:

- Use the Action Based Testing (ABT) methodology, a keyword-based, object-oriented approach that provides for visibility, reusability, scalability, and maintainability. Visibility, reusability, scalability, and maintainability translate into speed and cost savings.
- Choose the right enabling technologies that support the methodology. The tools need to support extensibility and a team-based global test automation framework, with a solid management and communication platform.
- Put in place the right people with the proper skills and training in methodology, tools, and domain knowledge (knowledge of the software to be tested, the industry for which the software is intended, and end-user expectations).
- Separate test design from test automation so that automation does not dominate test design. Action Based Testing (ABT) creates a hierarchical test development model that allows test engineers (domain experts who may not be skilled in coding) to focus on developing executable tests based on action keywords, while automation engineers (highly skilled technically but who may not be good at developing effective tests) focus on developing the low-level scripts that implement the keyword-based actions used by the test experts. ABT allows an organization to spend more time developing tests and less time actually coding the test cases. This speeds up the whole testing process and helps to reduce costs.
- Lower costs by using less expensive labor than your local team.

<p><b>Automated Testing: Productivity Objective</b> No more than 5% of the effort surrounding testing should be expended in automating the tests.</p>
---

- Jumpstart the process with a pre-trained outsourcing partner that knows more about test automation success than you do, and that has a competent, well-trained staff of software testers, automation engineers, test engineers, test leads and project managers.

The most essential element is methodology. The methodology is the foundation upon which everything else rests. The methodology drives tool selection and the rest of the automation process. The methodology also helps to drive the approach to offshoring the “appropriate” pieces of the testing process.

## **Executive Overview of Offshore Outsourcing**

Most companies are convinced of the need to offshore some or all of software testing. Offshoring offers the promise of significant cost savings. However, offshoring is more than simply moving existing software testing efforts to an offshore outsource partner. Executives making the decision to offshore testing must understand the possible pitfalls of outsourcing and offshoring, and must plan effective strategies to combat these pitfalls. Executives also must stay focused on the fact that an effective offshored testing effort is based on the strategic integration of methodology, the latest technologies, and an effective global resource strategy that supports the methodology and tools.

Some of the major pitfalls of offshoring include:

- Problematic communications due to language and cultural barriers, mismatched or miscommunication of expectations, poor metrics selection, and unresponsiveness.
- Insufficient or mismatched skill sets in the software testing organization, such as using entry-level development engineers as software testers, lack of knowledge of the software being tested, and lack of domain knowledge (knowledge in the category of the software being tested).
- Management issues due to the lack of a workable test management process and associated methodology.
- Vendor problems or vendor infrastructure problems such as poor data bandwidth.
- General offshoring risks such as security and protection of intellectual capital.

Some suggestions for dealing with these pitfalls include:

- Finding a trusted partner or building trust in a partner. You need to work with a partner that you know has testing experience, an experienced staff, an understanding of current methodologies, and competent domain knowledge.

- Train the test organization. Provide them with knowledge of your application, your expectations, your communication/management platforms and expected domain expertise. Discuss how to recognize and deal with cultural issues.
- Adopt a methodology and tools that support the overall methodology to improve testing, defect tracking, automation, and communications management, focusing on excellent and correct methods, ease of distributed team communication, accessibility, and useful measures.
- Choose carefully what work goes offshore and what remains “at home”. Often it makes sense to keep user-focused scenario development and business process testing at home where you have more knowledge of the domain and the user.
- Get someone local to manage the offshored test effort. A local lead that is part of your team, who understands the culture and communication nuances of the offshore team, can lead the project, effectively communicate progress and metrics, and help to streamline the process.

The most effective way to avoid the pitfalls of test automation and offshoring, and to realize the full time and cost saving benefits of test automation, is to implement a strategy of Global Test Automation.

Global Test Automation strategically integrates the latest test automation methodologies based on Action Based Testing, the latest testing technologies, with an effective and balanced global resource strategy that makes use of both onshore and less expensive offshore resources, on- and off-shore leads, and effective team-based management tools and methodologies.

The benefits of such an approach are scalability, reusability, visibility, and maintainability that ultimately will allow an organization to achieve both time and cost savings while delivering a quality software product. The bottom line is a higher quality product that is delivered faster and more cost effectively.

As you can see in Figure 3 below, Global Test Automation is the strategic integration of:

- Speed achieved through an ABT test methodology and test automation technologies for distributed teams
- Cost control achieved by using a cost structure based on world wide resources

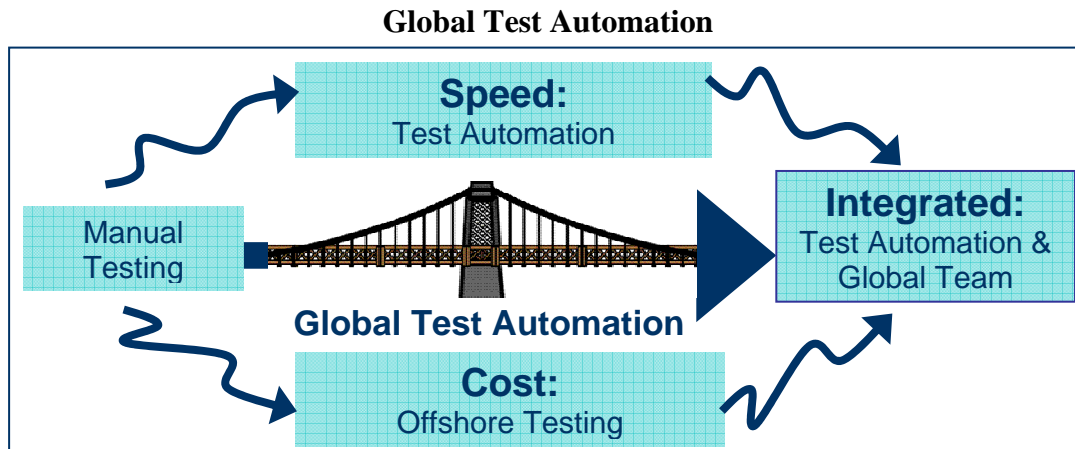


Figure 3

## Choosing a Global Test Automation Partner

Selecting the *right* testing partner to implement this strategy is more than simply selecting a vendor to develop and run test cases. It entails selecting a partner who fosters innovation to improve test productivity, a partner who understands Global Test Automation, a partner who understands that the methodology drives the use of appropriate tools, a partner who has an effective management and communication strategy, and a partner who makes use of global resources to provide the appropriate mix of on and offshore resources to drive down costs.

The process of test automation can be *jump started* with the selection of a pre-trained strategic partner that knows more about test automation success than you do, and that has a competent, well-trained staff of software testers, test leads, and an in-place global resourcing strategy.

**An innovative turnkey global test automation solution...**

LogiGear has the people, technology, and methods to help you *double* your test coverage, decrease testing time, improve product quality, and cut costs!

LogiGear is not only such a strategic partner; LogiGear represents a *best-of-breed* partner for software test automation. Some of LogiGear's strengths as a test automation partner include:

- Innovation - A thorough understanding of Global Test Automation integration and recognized industry thought leadership on the topics of Global Test Automation, Action Based Testing, and all aspects of software test automation. LogiGear's senior management and staff are extensively published, teach the topics at major universities, and regularly present at testing industry events.

- Transformation experience - Extensive experience implementing Global Test Automation with both small and large software development organizations.
- Test productivity optimization - A thorough understanding of the pitfalls and remedies for test automation.
- Software testing expertise - The ability to implement a global resourcing strategy that makes use of expert staff that is fully trained in all aspects of software testing, coupled with outsourcing locations that minimize cost and risk. LogiGear software testers, both on- and offshore, are all LogiGear employees and software testers by training. They receive extensive continuous skills development, and are efficient in the implementation of test tools.
- Global delivery services - The ability to implement a blended onshore/offshore test lead approach that helps to mitigate communications and cultural issues and improve both project management and reporting.
- Experience - Experience working in a globally distributed testing environment, including use of advanced automation for globally distributed teams.
- Collaboration for success - Experienced in the two-way exchange of methodology, tool, and management knowledge to the customer, and the transfer of product and domain knowledge to the offshore testers.

In LogiGear, you will be selecting a strategic test automation partner that will help you hit the ground running, implementing a global test automation solution that delivers *both* cost and time savings.