Minimize IT Outsourcing Risk with Collaborative Quality

Optimize the software lifecycle for greater quality and agility across distributed teams

February 2009

By John Michelsen and Jason English, iTKO LISA

 iTKO LISA
  1505 LBJ Freeway
  Suite 250
  Dallas, TX 75234
  USA

www:  http://www.itko.com
email:  info@itko.com
tel:    877-BUY-ITKO (289-4856)

©2009, Interactive TKO, Inc. All rights reserved.
Introduction

This decade has seen explosive growth in the practice of outsourcing IT to SIs (System Integrators) and services firms around the world. Initially, this trend was driven almost exclusively by decreasing the cost of development labor required to maintain companies’ increasing dependency on business technology. One specific area that quickly grew was the practice of outsourcing software testing to overseas teams.

Not every organization was able to realize the expected benefits. Enterprise systems are changing in important ways. They are becoming more complex and interconnected, as the huge monolithic applications give way to more componentized, services-based solutions. These decoupled apps offer more flexibility, but they also require new tools, methodologies, and expertise to effectively develop, test and deploy them across distributed teams. Failure to account for these and other factors resulted in disappointment as delivery schedules were put at risk, additional expense was required to ensure quality, and outsourcing relationships were strained.

Despite these challenges, outsourced IT functions still offer the potential to deliver significant competitive advantages to many organizations. Companies in North America and Europe continue to face a shortage of local talent. As the market for outsourcing matures, the leading global service providers are refining their IT practices and offering more value-added solutions to their customers. New markets for outsourcing are also emerging in China, Russia, and in “near-shore” Latin American countries to meet the demand.

An AMR analyst stated that companies could save up to 75% in costs(1) by outsourcing their software testing. However, simply calculating a reduced labor cost for manual testing does not provide a complete picture. There are many other factors for success, and a recent industry-wide survey from Aberdeen reported that 76% of North American companies cited that vendor management effort and costs of IT outsourcing were higher than expected.(3)

To prevent a disconnect between the IT service levels companies expect from outsourcing, and the quality of service they receive, companies should look at outsourcing as a symbiotic partnership that will benefit both the company and the SI, and align IT outsourcing capabilities to very specific business and software quality goals. This paper will focus on how companies can navigate the risks of collaborating across globally distributed IT teams for today’s complex software development and testing needs using iTKO LISA software tools along with embedding quality practices to better realize the benefits of IT outsourcing.

The Growing Outsourcing Trend

IT outsourcing has become a common practice for many firms. Gartner estimated that the volume of “mega-contracts” (more than a billion dollars) for outsourcing reached an all-time high in 2006, with more than US$31 Billion of these deal awards reported(4). The same study showed that the amount of mega-contracts dropped to around $15 Billion in 2007. However, overall the global outsourcing market continued a healthy growth rate of 10.2% through 2007(5).

Despite having to overcome challenges along the way, it is clear that today’s leading enterprises are well down the road of adopting IT outsourcing in one form or another. While each company has unique needs in terms of core development and testing practices, most outsourcing strategies are aligned to a few common models:
Common IT Outsourcing Models

1. Single Outsource Vendor. In this model, the buyer outsources all IT development and testing functions to a single SI partner. This approach provides the benefit of having a single source for delivery of needed business applications, and “one throat to choke” if something goes wrong. It can also create high switching costs if the decision is made to change vendors later.

2. Outsourced Testing. This model is the easiest route for many companies that decide to keep software development close to home. Since testing is by nature the process of “breaking” an application, many organizations find that a removed third party tester can be very valuable and impartial. This model was very easy to adopt when testing was a manual, point-and-click-on-the-screen endeavor that was best done by less costly staff. Another advantage is the time difference of having a testing partner in India, with development teams in the US or Europe, as the testing team can often conduct its activities and return results before the development team starts the following day’s work.

3. Development Support. The practice of outsourcing development took flight in India around the Y2K craze, as there was a limited pool of COBOL developers in the US who could make the many legacy system updates. Hiring and retaining skilled in-house development teams can be expensive. Many companies following this strategy are seeking both cost efficiency for non-differentiating development tasks, and access to developers with specialized skills needed to carry out specific tasks, such as migration between two different integration platforms.

4. Multi-Vendor. This represents a broader outsourcing strategy to save costs and access resources without being locked in to a single service provider. Commonly, development and testing activities will be outsourced to different SIs. In this model, the company hopes to keep costs down, while the multiple providers keep each other “in check” to ensure project success. This non-exclusive model can prevent some of the risks associated with a single outsource vendor, although it can also be harder to manage multiple firms and get them to collaborate as efficiently.

Understanding Outsourcing Risks

In today’s economy, outsourcing provides too much potential cost savings to be ignored. Yet, even though managing cost is obviously important, avoiding risk is even more important.

A lot of companies in the early stages of offshoring failed to recognize that a $33/hour rate in India (for example), versus $100/hour in the US, doesn’t simply translate into a linear two-thirds development cost reduction for the overall project. A successful project goes beyond delivering code that compiles before the deadline. Success needs to include a set of quality criteria that need to be managed by both the outsourcing consumer and provider. After all, what good is a lower cost project if it takes 3 times as long to repair, or if you risk failure in front of customers? Your outsourcing partner needs to be incented to build quality into the application, and it’s equally important that an IT infrastructure is in place to facilitate and manage quality across the lifecycle.

“Although user organizations often have fundamentally sound procurement organizations to initiate outsourcing contracts, for many, their IT sourcing strategies and governance structures are still immature, lacking altogether, or misaligned with enterprise objectives. Because these organizations lack the basic building blocks for successful vendor management and outsourcing success, expected cost savings and other benefits are difficult to obtain. In extreme cases, the lack of needed trust and control to optimize the outsourcing relationship results in deal failure.” - Gartner (6)

What are some of the challenges and risks associated with outsourcing software development and testing across distributed teams? Of course, time zones, language and culture can all
represent significant collaboration obstacles -- careful outsource partner selection and good management practices can help avoid or alleviate those issues. But what about other inherent difficulties associated with structuring work across multiple teams, keeping those teams highly productive, and assuring that their deliverables meet the larger requirements of a complex, distributed system?

**Managing accountability across multiple teams.**
Many larger companies have a multi-vendor strategy of procuring IT services. However, the more companies and teams that are involved in a project, the more opportunities there are for information sharing and collaboration to break down. Collaboration problems can develop, not just between outsourcing partners, but also with internal teams, and finding the root cause of these issues can be difficult without a high amount of management involvement. What is needed is greater team accountability for shared deliverables all the way across the delivery lifecycle, which includes visibility into problems early, versus later in the deployment where they become significantly more costly.

**Dependency on unavailable or inaccessible systems.**
Interdependent teams working together can be constrained by limited access to live applications and components that are not yet completed or available for testing. When there is increased contention for shared resources, agility suffers as teams are forced to queue up for access, introducing scheduling delays and project risks. In some cases, teams may attempt to maintain their own test environments by replicating the components they need in a testing lab or custom test harness. This can become very costly to manage, and requires a high level of configuration, licensing costs and maintenance to keep current, even if it is running on virtualized hardware (which also has incremental licensing costs). And of course, some sensitive systems such as ERP mainframes are simply too big and have too much overhead to be replicated by conventional hardware virtualization means.

**Solution: Collaborative Quality Across Distributed Teams**
Successfully addressing these risks and challenges can unlock the larger benefits and potential of outsourced IT initiatives. If the appropriate IT infrastructure was available to better manage accountability and quality across distributed teams, while eliminating the dependencies and constraints that restrict the agility and productivity of these teams, then it follows that these benefits could apply not just to outsourcing, but to any distributed team environment.

**How iTKO LISA Supports Collaborative Quality**
In order to trust that every IT function assigned to a services partner will meet the needs of business, we must first commit them to deliver components against well-defined requirements, in an auditable form, with the ability to accommodate the highly interdependent nature of today’s applications. This requires not just careful management of the outsourcing vendor – but the appropriate resources, methods and tools to accomplish the job.

iTKO’s LISA product suite provides key testing, validation and virtualization capabilities that will help firms mitigate these risks, and foster more collaborate quality across interdependent teams:
1. A collaborative testing framework that directly verifies all technology layers with a high level of automation, so the team can better meet defined requirements at lower cost;

2. Continuous Validation so both parties can assure that each software component supports all upstream and downstream dependencies, in an auditable form; and

3. The LISA Virtual Service Environment, which simulates the behavior of the highly interdependent and critical systems the teams must build and test against, even when they may not have access to live or completed components.

With these capabilities, the consumer of services receives not just code assets, but also the deliverables to rapidly validate them. iTKO LISA helps commit those distributed teams to not only deliver compiled code and test scripts on time, but also to ensure that they are delivering quality into the lifecycle.

Collaborative Testing Aligned to Functional Requirements

Resolving a misunderstanding early in the requirements phase can be orders of magnitude less expensive than fixing it later in deployment. However, testing often gets short shrift when up against a tight deadline. Often testing happens at the end of delivery, when there is already a user interface defined. How can we move testing earlier into the lifecycle, to save costs and improve quality?

If the offshore partner’s testing is not tied to functional requirements, there is no way to know if there is any value to the test. To try to test earlier in development, many coders write unit tests, which are useful for very fine-grained checks during development. However, if the partner claims to offer 90% test coverage with a patchwork of unit tests, the consuming organization would have to be able to audit all that Java code to make sure it is verifying the functionality. Furthermore, it is inevitable that developers will test what they think the application should do, rather than what it should actually do, which of course is the primary reason QA practices exist. What is needed is testing not only in development, but across the lifecycle and closer to those understanding the business needs.

There are several leading ALM (Application Lifecycle Management) and Test Management vendor suites in existence that provide a great foundation for managing and tracking requirements across the team, such as Borland, Rally, MKS, Rational, HP TestDirector for QualityCenter, and others. Often the requirements or tests defined within these tools are simply text descriptions or spreadsheets with the steps of the intended functionality. This is an important first step, but it is not enough.
The key to achieving collaboration is to demand that each component a team delivers is also delivered with a suite of working tests that prove its functional requirements have been met. That way the next team, whether local or offshore, can iterate on testing from there. LISA tests, saved as .XML documents, can augment these processes by being combined with traditional requirements docs as executable test assets in ALM and Test Management tools.

From this foundation, LISA provides a “codeless” way to rapidly build and execute functional tests that can directly invoke and verify each component, so that it can be proven to match defined requirements in later testing cycles. QA and business analysts can reuse the tests and expand them, without coding, to account for the needed use cases. LISA leverages a declarative testing approach that is more portable, so that test cases do not break when data is changed in the middle tier or other changes are made in upstream or downstream dependencies.

**Continuously Validate Changing Components and Dependencies**

In a service-oriented world, a change to any service or component can create “unintended consequences” by causing unexpected failures on other dependent services and workflows. These risks are compounded when third parties and distributed teams are involved in providing or consuming them. What is needed is a way to continuously validate that delivered components have accounted for other elements in the workflow, and declare those in the tests, providing other team members with visibility into those dependencies.

Take the example of an outsourced development partner that is integrating their component to a data source, or a third party component without your knowledge. The developer may simply think that if it is available as a service, it is free to consume. However, to prevent unintended consequences in the future, the developer must test and declare the dependency in the context of my use case. Without this, if the software doesn’t function in the future, all we know is that SOMEONE didn’t deliver – but we lack the visibility to find the source of the unintended consequence.

LISA can be used to “coat dependencies with testing” from this perspective, to prevent unintended consequences in the first place. LISA can be invoked to run automated tests from the server, either on a scheduled basis, as part of a build process or in Service promotion activities with leading SOA registries. When a development team checks in a new build or service version, LISA can run suites of tests; then communicate data, metrics and alerts back to other teams and processes along the dependency chain.

As a best practice, teams often use LISA to segment continuous validation activities along business process lines. For example, developers could be creating very granular functional tests for their delivered components, and those are checked in to a workflow test which is refined and run by the QA team. A business team on the client side may take all of those tests and run them continuously to check the integrity of the overall business process every day, or as frequently as desired.
Enable Parallel Development and Testing without Constraints

Distributed teams need to be able to decouple their own environments from dependencies on live or incomplete systems, so they can continue working in parallel, rather than waiting on each other. A large enterprise system may leverage 100 or more separately developed services and components, which are highly dependent and interconnected to deliver the desired business functionality. The teams building these underlying components must do so in relation to the rest of the larger system, but often with limited access to live systems and data. Many times, getting a custom test harness or hardware-based test lab to replicate all of the layers in these distributed, multi-tier environments for testing purposes is simply too costly and difficult to support.

LISA Virtualize solves this problem by allowing teams to design, develop and test their own components against changing environments by capturing and modeling the rest of the system (services, data, mainframes, etc) as running Virtual Services. This robust behavioral simulation of these testing assets and their underlying implementation layers can also be employed as a ready target for load and performance testing. By using Virtual Service Environments (VSE), multiple teams can design and test services in parallel, eliminating the constraints and dependencies of accessibility, availability and security for incomplete components and live service implementations.

For instance, consider the challenge of ensuring performance in a financial "rate quoting" application, that is consuming three separate service components (shown as Verify, Lookup, and Response), developed by three separate teams. Both functionally and at a performance level, your business unit is signed up to make sure that the delivered application is meeting a defined Service Level Agreements (SLAs) of 2.1 second response times.

In order to achieve the SLAs above, each service provider team must ensure their own response time SLAs at the component level. The only way the "Response" provider can deliver with certainty is if they can test their own response times without being dependent upon the other service providers and your consuming application. By virtualizing away the rest of the architecture, the team creating the "Response" service can test against a realistic model of the system throughout development, and commit to their own response time of 1.2 seconds with much more certainty that it will perform the same when consumed by the completed application.

This practice of Service Virtualization can have a significant positive impact on the productivity of distributed development and testing teams. With VSE, everyone can increase agility and move toward quality delivery, without stepping on each other’s toes (for more information visit: http://www.itko.com/sov).

Best Practices for the Outsourcing Consumer

We’ve discussed key capabilities that can make collaborative quality far more effective for distributed teams, including outsourced ones. Now let’s summarize the corresponding best
practices that the outsourcing consumer or buyer can implement to generate maximum value to their organization.

**Reward Outsource Partners for Delivery Success… Not Just Delivering.**

There is risk in prematurely “signing off” on delivered code assets before adequate testing to ensure requirements are met. It’s important to verify delivered components meet functional requirements within the context of the larger system implementation. LISA provides these capabilities and integrates into most IT management tools of choice, so business teams can use LISA tests as an automated, executable, and enforceable means to measure project success and ensure quality across the development lifecycle and distributed teams.

**Trust, But Verify… Continuously**

As a consumer of offshore services, we need to bear the responsibility of ensuring that each piece of code that we integrate into our applications does not trigger unintended consequences. This requires distributed teams to document component dependencies and test them as part of their deliverable. The consumer is then responsible for taking those tests and adding more tests at a business process level to ensure that the expected behaviors are verified in the context of the consuming workflow.

Using LISA’s no-code environment, the QA team can rapidly build larger business process tests and validations, which reuse the underlying component-level tests as “sub-processes” within that workflow, allowing QA teams to focus more on the overall process.

The QA team does not need to know about every possible side effect of underlying components to manage the tests, but if an unintended effect occurs, LISA can pull up the root cause, which can rapidly be communicated as an executable test asset for the responsible development team to resolve. LISA was designed to accomplish this level of Continuous Validation for distributed teams, and provides the automation needed to validate the entire business workflow at build time, change time, and runtime.

**Enable Greater Productivity by Removing Constraints**

Even the most efficient and experienced development teams will be hampered by restricted access to live services and systems needed for testing. Service Virtualization can enable multiple teams to work freely in parallel, rather than at odds due to the constraint of having access to live or incomplete systems. LISA Virtualize provides your partners with a way to decouple themselves from dependencies, by modeling test resources as Virtual Services that behave just like the rest of the application for testing and development purposes. Project risk is also reduced as each team is able to perform tests much earlier in the development and delivery lifecycle, without impacting sensitive systems and data.
Conclusion

A partnership between outsource consumers and outsource providers based on shared objectives for collaborative quality are far more likely to deliver the benefits and results expected by outsourcing. Success depends upon a combination of people, processes and enabling technology to foster greater productivity, agility and quality across distributed teams. Key capabilities to realize these objectives include a collaborative framework for testing functional requirements, continuous test and validation of delivered components within the context of larger systems, and virtualization of constrained resources for greater productivity and improved time-to-market for distributed teams.

The benefits of implementing these capabilities are not limited to outsourced IT providers, but are applicable to any distributed development or testing environment consisting of multiple teams, whether offshore, on-shore, or in-house. Similarly, these capabilities can deliver greater value across the software development lifecycle, with benefits increasing as collaboration is promoted between development, QA, business analysts, and other stakeholders in the software delivery process.

By implementing the infrastructure to support collaborative quality and sharing the responsibility for quality with an outsourced partner, the buyer of outsourced services can both reduce the cost and risk of software development, and realize the benefits expected from IT outsourcing.

References


6. Ibid.
About the Authors

**John Michelsen, Founder & Chief Architect, iTKO LISA**

John has over fifteen years of experience as a technical leader at all organization levels, designing, developing, and managing large-scale, object-oriented solutions in traditional and network architectures. He is the chief architect of iTKO's LISA product and a leading industry advocate for software quality and virtualization.

Before forming iTKO, Michelsen was Director of Development at Trilogy Inc., and VP of Development at AGENCY.COM. He has been titled Chief Technical Architect at companies like Raima, Sabre, and Xerox while performing as a consultant. Through work with clients like Cendant Financial, Microsoft, American Airlines, Union Pacific and Nielsen Market Research, John has deployed solutions using technologies from the mainframe to the handheld device.

**Jason English, VP Communications, iTKO LISA**

Jason brings more than 14 years of experience in executing marketing plans, re-engineering business processes and meeting customer requirements for technology and consumer companies such as HP, IBM, EDS, Delphi, TaylorMade, Sun, Realm, Adaptec, Motorola and Sprint. As Director of eMarketing and Executive Producer of i2’s elite in2action interactive consulting business unit at i2 Technologies, he was responsible for i2's outbound messaging during a period of extreme growth, as well as marketing services and working directly with clients to build easy-to-learn workflows and front-ends to B2B collaboration systems. Prior to that, he served as one of the first "Information Architects," managing customer experience for Fortune 500 clients at Agency.com. He has also designed several internationally released computer games in addition to conventional print advertising and television commercials.

About iTKO LISA

iTKO helps our customers transform the software development and testing lifecycle for greater quality and agility in an environment of constant change. iTKO’s award winning LISA product suite can dramatically lower quality assurance costs, shorten release cycles, reduce risks, and eliminate critical development and testing constraints by virtualizing IT resources to provide accessibility, capacity and security as needed across interdependent teams.

LISA enables test, validation, and virtualization solutions optimized for distributed, multi-tier applications that leverage SOA, BPM, integration suites, and ESBs. iTKO customers include industry leaders such as eBay, American Airlines, Allstate, Time Warner, SwissRe, Bank of America and government agencies including the U.S. Department of Defense.

For more information on iTKO LISA solutions, visit [http://www.itko.com/lisa](http://www.itko.com/lisa), contact iTKO at info@itko.com or call 1-877-BUY-ITKO (289-4856).