THE THREE PERILS OF SOFTWARE DEVELOPMENT:

Understanding and Overcoming the Root Causes of Software Failure in eCommerce

JULIE HAGAN
Senior Practice Manager

EXECUTIVE OVERVIEW

In the competitive world of Online Retail/ eCommerce, your customers have all the power—and their expectations are as clear as their loyalty is fickle. If you're not providing a seamless, hassle-free user experience, you're losing them to someone who is.

To your users, every defective line of code is a customer service failure, every unexpected crash an abandoned cart, and every delayed update only puts you further behind the competition. Your peers understand this well. **That's why they've increased their software testing and QA spending by 43% over the last 3 years.**[1]

Unfortunately, major software projects face an uphill battle from the start. As any developer, IT leader, or business executive can attest, major software projects face numerous obstacles en route to release. From slight missteps to major failures, these challenges threaten to derail the initiative.

Ultimately, as costs overrun and the release date slips, the business and IT soon find themselves in a precarious situation: reduce the project's scope, cancel it altogether, or accept the ensuing financial fallout of letting the embattled development run its originally-intended course.

Outcomes like that aren't isolated, either. In fact, of all major software projects, one in six will suffer dramatic schedule and budget overruns—or what's also known as a "Black Swan". **With an average cost overrun of 200%, and an average schedule overrun of almost 70%[2], "Black Swans" don't just threaten the credibility of business and IT leaders; they can put entire departments years behind.**

In an industry that moves as fast as eCommerce, the opportunity loss of falling behind the competition is can be especially detrimental. In fact, sometimes it can even be more costly than a busted budget.

It's tempting to ascribe these failures to poor planning and project management, but that ignores the root causes of these problems—**The 3 Perils of Software Development™: Defects, Delays, and Dollars.** Together, these often-overlooked setbacks combine to sink the anticipated ROI of your software initiatives—plunging them into the perilous waters of cost and schedule overruns, poor-quality releases, project cancellations, and more.

When coupled with heavy competition, near-constant disruption/innovation, and high customer expectation, the 3 Perils become even worse. Black Swans are never routine, but in the retail industry, they can be especially crippling—transforming into credibility-killing disasters that jeopardize future revenue and drive your customers into the waiting arms of the competition.

In this white paper, you'll learn what you need to know about the 3 Perils: what causes them, the problems they create, and—most importantly—what you need to do to ensure that your next major software project is a success.

[1]World Quality Report, 2016

[2]Harvard Business Review, 2011

INTRODUCTION

No matter the size of the undertaking, successfully developing and launching a software initiative is not a process to be taken lightly—especially within the highly-competitive Online Retail industry. The road to release is treacherous—littered with countless obstacles and challenges (both visible and hidden) that threaten to derail your project and jeopardize your company's monetary investment.

This isn't groundbreaking or revelatory. But for such a well-known fact, history continues to repeat itself and software projects continue to fail—causing irreparable damage to the credibility of IT and business leaders alike while fracturing customer loyalty and squandering corporate capital in the process.

THE SNOWBALL EFFECT

Many companies describe a similar experience: a prolonged development lifecycle rife with missteps and setbacks. Business and IT alignment might have existed during the initial phases of development, but this quickly fades as complications arise in both camps. Whether these problems are from IT errors or the downstream impact of what the business thinks are "small changes" is immaterial. For every step closer to release the project gets, development teams risk taking two steps backward as a slew of minor issues combine to create major rework and delays.

While these setbacks may initially appear insignificant when viewed individually, they present a far larger problem as a whole. As these complications aggregate, they don't merely generate costly rework for IT teams; the inflated costs and lengthy implementation delays can quickly metastasize into a much larger investment than business and IT leaders had originally forecast—leading to cost-cutting compromises such as scaling back the project's scope or cancelling it altogether.

In fact, of all major software projects, only a mere 6% are considered "successful" upon their release—coming in on time, on budget, and within scope. Of the remaining 94%, more than half miss on at least one of those three criteria (for example, coming in on time and on budget, but with drastically-reduced scope) while the remaining projects are cancelled altogether.[3]

"OF ALL MAJOR SOFTWARE PROJECTS, ONLY 6% WILL RELEASE ON TIME, ON BUDGET, AND WITHIN SCOPE"

[3]Standish Group, 2012

To today's online retailer, those numbers are unacceptable. In an industry defined by user experiences, one cannot afford to risk 94% of their projects going off the rails.

In the face of these high-profile project failures, users have made their expectations exceedingly clear. **In fact, 86% have reported that the speed and quality of a company's website or mobile app affects their trust of that company overall[4] —and 89% have indicated that they will go to a competitor's site following a poor user experience.[5]**

Unfortunately, in spite of such clear consequences, many companies still have difficulty finding and correcting the root of their problems. While this is understandable (it's no small task), in an industry that places a high premium on quality, it's a necessary activity to undertake.

THE 3 PERILS OF SOFTWARE DEVELOPMENT™

In truth, almost all software project setbacks are borne of one of three sources—the 3 Perils of Software Development™: Defects, Delays, and Dollars. Understanding each of their various causes is integral towards safeguarding your next software development project.

Defects

Costing 5x more to fix over each stage that they lie undetected, defects cripple productivity, infuriate users, and generate costly rework for IT teams.

Delays

Lengthy implementation delays hold up valuable releases while eroding department credibility.

Dollars

Mounting defects and delays run up costs—leading to busted budgets, reductions in scale, and project cancellations.

DEFECTS – THE FIRST PERIL

Perpetually present, defects lie hidden beneath the surface of your code—waiting to wreak havoc upon software projects. Like other software setbacks, defects are oftentimes misunderstood as relatively insignificant problems. This couldn't be further from the truth. Where a few low, or even medium-priority defects will never make or break a software system, a high number found late in the SDLC can be disastrous—inflating costs, extending schedules, and causing quality to nosedive.

DEFECTS BY THE NUMBERS

5X – How much defects increase in cost by over each SDLC stage they go undetected

10,000 – The amount of defects Lighthouse has seen in a client's production-level software system

$3,000,000 – The total rework cost of correcting as little as 100 production-level defects

[4]Dyn, 2015

[5]RightNow/Nielson Group, 2011

Few companies rely upon their software systems as much as online retailers. With POS/Inventory-Management systems on the back end and user-facing mobile apps and eCommerce sites on the front, the stakes are higher than they've ever been.

In an industry defined by high competition and exacting customer standards, a high number of production-level defects can have serious consequences. With high levels of innovation and disruption, everyone is fighting neck and neck to outdo each other by creating the best possible user experience for their customers. In fact, by 2020, they are expected to overtake price and products as a key brand differentiator.[6]

When done right, high-quality user experiences can be a boon for loyalty—and will keep customers coming back again and again. But when done poorly, defect-laden websites and mobile apps work the exact opposite way—jeopardizing future revenue while driving customers into the competition's open arms.

It's well-known that customer loyalty is critical to online retailers. But while users have no problem judging entire companies by the quality and performance of their websites and apps—and engaging competitors when their expectations aren't met—their loyalty is also quite costly to replace. **In fact, a company will pay 6-7 times more to acquire a new customer than they would to retain an existing one.[7]**

[6]Frost & Sullivan, 2015

And that's just the tip of the iceberg. Unfortunately, software defects can take a costly toll on your IT budget, too.

Across each development stage it goes undetected, a defect becomes five times costlier to correct. **That means a production-level defect will cost a company as much a $30,000 more than if it was identified in requirements!**

For example, a well-known national retailer was looking to close the increasingly large gap between them and their competitors when they kicked off a $1.4 billion IT modernization project in 2000. By 2001, they'd realized that their new system was prohibitively expensive, so they launched a smaller $600 million project to upgrade their supply-chain management software. Within a year, that project had gone off the rails as well, and by 2002 the company had filed for bankruptcy.[8]

Cautionary tales like this have led IT leaders and business executives alike to an inevitable conclusion: **software quality is essential and must be safeguarded.**

WHERE DEFECTS COME FROM

While it's easy to identify the developer who introduced the error as the root cause of a defect, it's unreasonable to expect they will

[7]Bain & Company, Date Unknown

[8]Harvard Business Review, 2011

ever be able to deliver completely error-free code. Instead, it's wiser to accept that defects will always be present in code, and to focus instead on how and why they regularly make it into production undetected.

So how does that happen? That answer is as diverse as one would expect, but here are a few of the most common causes:

INADEQUATE TESTING

As the final stage of development before going live, testing is the last line of defense to cleanse coding errors. Unfortunately for a lot of companies, a majority of defects manage to slip through undetected.

Whether it's due to a lack of resources, budgetary or scheduling constraints, inadequate testing environments, developers testing their own code, or myriad other reasons, testing is often compromised due to its perceived "lack of necessity" in the development process.

POOR REQUIREMENTS

Of all sources of defects, poorly-defined requirements are one of the most common. Few companies have stringent enough controls in place to prevent vaguely-worded requirements—which are then left in the hands of a developer to interpret.

Without specifically-defined instructions, the coder has innumerable ways to accomplish their task—even though all but one of them will lead to an incorrect outcome.

In eCommerce, poorly-defined requirements can be especially costly; as a single misinterpretation can adversely affect user experiences—to the detriment of future revenue and customer loyalty.

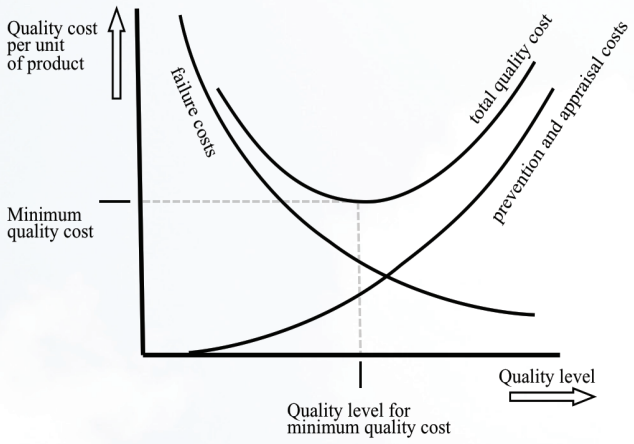
BAD EXIT CRITERIA

Like all things, quality exists on a continuum. Test too little, and bug-laden, frustrating software is the result—jeopardizing future revenues by driving customers away; test too much, and a company wastes spending on diminishing returns.

Bearing that in mind, it's unsurprising that the business often doesn't share the same standards of quality as IT. In most cases, this leads the business to push IT to ignore quality and drive toward the scheduled release date—resulting in reduced testing, and the reclassification of critical defects to get around rework.

The end result is similar to a new car left outside in a hailstorm: brand new, but already bearing the scars from thousands of dings and imperfections.

Total Cost of Quality



UTILIZATION OF NON-PROFESSIONAL TESTERS

Like newspaper editors, testers need to be truly independent of the development team to provide objective, bias-free results. Unfortunately, many IT organizations lack the specialized personnel to carry out testing tasks—oftentimes resulting in testing being carried out by inexperienced (and costly) subject-matter experts (SMEs) or the developers themselves. This also introduces the potential for bias in the testing process, since developers will often miss their own coding errors.

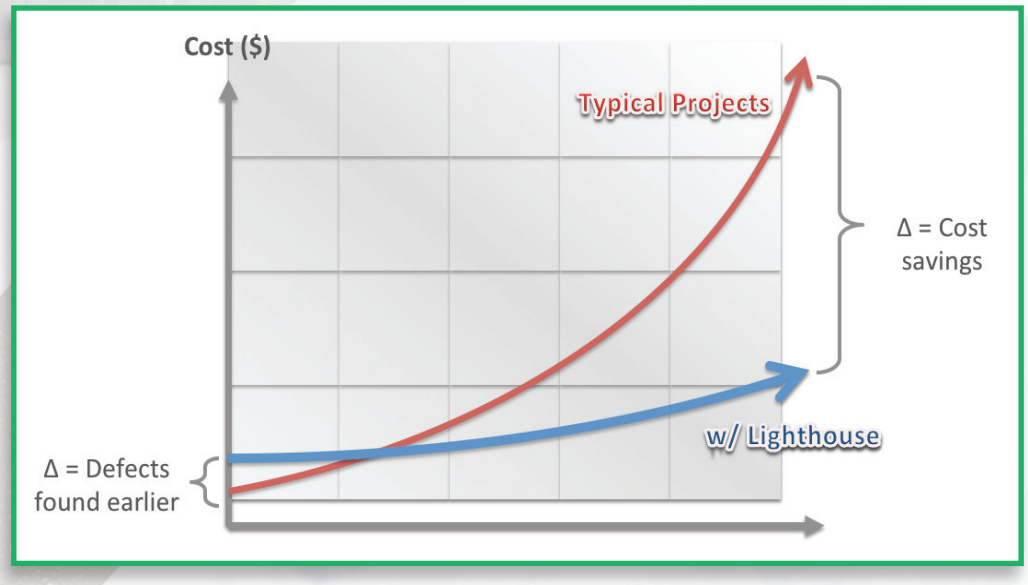
PREVENTING DEFECTS IN THE LONG TERM

Reversing these outcomes takes commitment.
Fortunately, there are a number of efficient ways to do so:

HIRING A THIRD-PARTY OUTSOURCER/CONSULTANT

Since most companies with these problems lack a culture of quality within their IT organizations, self-improvement can be difficult and prohibitively expensive. That's why third-party vendors are generally a far more efficient source of improvement.

Companies like Lighthouse Technologies, Inc. can offer everything from full-scale software testing outsourcing to Quality Assurance assessments and consulting, helping ensure that no matter what a company's need may be—whether it's full-service testing or simply implementing the processes and tools to create a culture of quality—it's met by an expert team with the industry experience and technical expertise to help ensure the best possible result.



PEOPLE FIRST, THEN PROCESSES AND TOOLS

When faced with improving their organization's quality, many IT leaders are tempted to take the "Implement & Install" route: implementing the best processes, installing the best tools, and having faith that their personnel will keep up. But while this may initially look good to upper management, it won't produce the results they expect.

Instead, it's more beneficial to adopt a "People First" mentality. The same methodology that Lighthouse Technologies, Inc. employs, the first step toward improvement should always be the hiring and training of dedicated testing personnel—instead of relying upon SMEs and developers. While they sometimes possess enough familiarity with native software systems to be effective, they're not scalable.

Once a dedicated testing team is in place (either outsourced or in house), they should be trained to follow defined, repeatable processes (for example, user stories/ requirements inspections help avoid costly rework and delays) while the organization supplements their skills with the proper tools when applicable (like automation, which increases overall test coverage and efficiency).

MEASURE EVERYTHING

There's a reason that the most advanced testing organizations in the world measure everything—it's the key to long-term improvement.

Metrics are more than mere data; they're an insight into every phase of an organization's development process. In terms of defect-elimination, no metric is more valuable than defect removal efficiency (DRE), which tracks the total number of defects found and removed during development.

When tracked over every phase of the SDLC, DRE can pinpoint the exact location where defects originate—enabling organizations to identify their weaknesses and adjust accordingly. While most companies are at or around 80% overall, that number is too low for eCommerce companies, and leaves their systems susceptible to defects and expensive rework. That's why Lighthouse strives for 95% DRE in every engagement, because it is the ideal balance of quality and economy for most Online Retail software systems.

DELAYS – THE SECOND PERIL

If production-level defects are the lightning of software development projects, delays are the thunder, following closely afterward and with startling effect. And even though the disastrous effect they can have on software projects has been well documented over the years, they're often dismissed as insignificant "setbacks" along the road to release.

Just like defects, delays typically fly under the radar until the damage they cause is too problematic to be ignored. In terms of software, this generally happens when the business realizes that rework (often spent correcting late-stage defects wrought by poor requirements) and other unplanned changes have taken their toll and made the anticipated release date little more than fantasy. At this point, one of two outcomes is likely to occur:

1 The business directs IT to push toward the release date, regardless of the repercussions.

As frustrations mount and the anticipated release date slips, the business presses IT to get the release out as soon as possible. The result, a development team myopically focused on getting the project out the door, is rarely a good one—as the team cuts corners to placate the pressing schedule. Unfortunately, this also adversely affects the software's quality and scope—as remaining work is jettisoned to ensure a (relatively) timely release.

It's a no-win situation for IT; as the defect-laden release will frustrate users (and threaten their loyalty) while still not escaping the ire of the business (who won't be pleased by the reduced scope).

2 The development team presses forward without making adjustments.

While a myopic focus toward a release date may be a recipe for disaster, ignoring delays and letting them run their course is no better.

If delays have accumulated enough to push back the release date, it's likely that the problem is systemic (poor/changing requirements, underestimated work effort, etc.) and that blindly pressing forward will result in more delays—racking up more expenses while pushing the project further and further off course.

As an online retailer, neither of these outcomes are ideal. Driving toward the release date with zero thought or concern for the quality of the end product will do nothing but alienate your customers; but incurring lengthy delays only puts you further behind the competition.

With these consequences looming, understanding the root causes of delays is imperative to fostering long-term viability for software development projects, IT departments, and the business.

DELAYS BY THE NUMBERS

10,436 Test Cases:
Total number of test cases Lighthouse ran on a client's new eCommerce site

939 Defects:
Number of unique defects found—over 5x more than anticipated (their offshore developer claimed to be CMMI Level 5)

1 Year:
The length of the delay to complete the rework—causing the client to miss their original holiday deadline and launch the next year

WHERE DELAYS COME FROM

For some organizations, schedule overruns are a regular occurrence. Despite vigorous pre-planning, the road to release is a rocky one and projects rarely come in on time. Whether it's three days or three months, delays can come from just about anywhere.

Like defects, there are myriad causes that can conspire to push projects off-schedule. Listed here are a few of their most common—and most overlooked—causes:

DOWNSTREAM IMPACT OF SEEMINGLY "SMALL CHANGES"

It doesn't matter if they're new to their position or a 20-year veteran, every IT leader has fielded requests from business leaders for new features or functionalities. But these "small changes" are anything but—and often have major impacts on project's schedule.

New bells and whistles are to be expected on an eCommerce development (as they're essential building blocks for a good customer experience), but since they weren't part of the original requirements process, they need to be created from scratch—a task that requires serious planning, coding, testing, and rework.

Changes like these are some of the strongest head winds that development projects face—always threatening to blow the initiative off its intended course. They're often thought to be a problem faced solely in Waterfall developments; but it's still a concern for Agile projects as well. Even though the process is iterative and change is expected, scope must still be carefully managed throughout the process to prevent schedule overruns.

POOR PLANNING AND COORDINATION

Major eCommerce/Online Retail projects are the sum of many lesser, independently-moving parts—oftentimes involving teams fielded by IT, the business, and third-party vendors.

Without historical data and metrics, accurately planning the schedule for of each of these various parts is an exercise in futility—rarely amounting to more than a "best guess".

DEFECT-RELATED REWORK

The First Peril may be a large enough problem on its own, but the effect that it has on the Second Peril shouldn't be discounted. Rework caused by a high number of defects can be more than enough to push a project off schedule.

PREVENTING DELAYS IN THE LONG-TERM

Just like preventing defects, there aren't many "quick fixes" here. These changes are as much cultural as they are process-oriented. Fortunately, the cost savings they offer is more than worth the effort they require.

INSPECT REQUIREMENTS

Having testing/QA personnel review requirements after they've been captured is invaluable towards preventing costly defect-related rework and delays.

In doing so, problem areas like vague and misleading wording can be caught before developers misinterpret them and introduce needless defects into their code.

IMPROVE TESTING PLAN

Without a repeatable, metrics-based testing plan, software testers have no exit criteria upon which to assess if testing is truly finished. Without a clearly-defined end point, testers often find themselves testing too much or too little—which can make staying on schedule near impossible.

TRAIN DEVELOPERS

Of all the changes listed, this may well be the easiest to implement. Whether they're in-house, outsourced, or staffed by a third party, requiring a Tech Check™ score of at least 80 for all developers on staff is an easy way to reduce defects and rework.

Improving a developer's score from 60 (the average score of a normal, non-specialized developer) to 80 not only improves their productivity; it reduces the total defects they introduce by 50%. This helps to reduce delays and prevent unsatisfactory customer experiences.

That's why predictive tools like Lighthouse's Code Analyzer and Early Detection Process (core tenants of our proprietary True North Testing Methodology) are so important. By utilizing them, testers know exactly how many defects they expect to find in each stage of testing.

Along with providing a clear end point, this also identifies risk areas, enables the test team to ramp up and down according to the number of defects they've identified, and—most importantly—avoids the rework and delays associated with under-testing and over-testing.

Ultimately, as your organization grows more mature in preventing delays by tracking metrics, you won't just notice your projects coming in on time more and more often, you'll also be completing them faster than you ever have before!

DOLLARS – THE THIRD PERIL

Dollars, the third and most serious Peril of Software Development, is one online retailers know all too well.

Every year, an estimated $300 billion is lost globally to poor user experiences—2/3 of which winds up getting spent with competitors.[9] But that's just a drop in the bucket when compared to the approximately $4 trillion in merchandise that's abandoned in shopping carts every year.[10]

Dollars add up quick in development as well, and can lay waste to software initiatives when they do.

Take the rework caused by defects, add the lengthy schedule overruns wrought by delays, and the resulting busted budgets and financial losses are too much for most companies to overcome.

DOLLARS BY THE NUMBERS

$2 Billion
Amount invested by a national retailer over a 2-year period into an IT modernization project

67,000
How many employees lost their jobs when the retailer declared bankruptcy after the project failed

600
The amount of stores the retailer closed before they were acquired by another corporation

[9]Frost & Sullivan, 2016

[10]BI Intelligence, 2014

As defects and delays continue to drive up costs, the anticipated Return-On-Investment (ROI) that originally justified the initiative's expense begins to quietly erode away. Like any slow decay, the results may not be immediately realized, but once they are things will move fast. With no end in sight, business leaders are faced with three options: continue the project and lose more money, reduce scale, or pull the plug entirely.

On average, major software initiatives will overrun their budget by 27%, while one in every six will become an aforementioned "Black Swan"—overrunning its schedule by an average of 70% and its budget by an average of 200%.[11]

Failures of that magnitude can cost executives their jobs and, in some cases, destabilize the financial viability of the company itself. And as worldwide investment continues pouring into large-scale software initiatives, the need for preventative measures will continue to grow more and more imperative.

PREVENTING DOLLARS FROM SINKING YOUR PROJECTS

While the most obvious way to ensure the long-term viability of software investments has already been discussed (the aforementioned strategies for preventing defects and delays), there are a few other wider-reaching strategies that can help prevent the 3 Perils from undermining your software initiatives.

TEST EARLY AND OFTEN

It's already been discussed at-length, but the necessity of high-quality testing cannot be overstated. Defects are a major factor in rework; which itself is a major contributor to the implementation delays that so frequently mar software projects—both of which drive up costs and threaten to derail your projects.

While fielding a team of professional testers is good, getting them to implement regular requirements inspections is even better. And equipping them with the predictive tools that enable them to do their job faster and more efficiently is best.

AUTOMATE WHEREVER POSSIBLE

Whether it's a mobile app or back-end inventory-management system, complex software systems all suffer from the same problem: the necessity for SME involvement in the testing process. It's as much a strain on the business, who'd rather have their personnel utilizing their valuable time on profit-bearing initiatives, as it is on IT departments, who'd rather have testers that answer to them and their own priorities.

For companies in these kinds of situations, test automation is the ideal solution. With an automation expert like Lighthouse Technologies, Inc., IT departments can dramatically increase both their test coverage and testing speed, while cutting their testing budget in half!

[11]Frost & Sullivan, 2015

MINIMIZE HIDDEN COSTS FROM VENDORS–ESPECIALLY OFFSHORE

Companies who utilize multiple third-party vendors (especially offshore teams) would do well to examine their collaborative relationships to see if they're losing money to "churn"—lost productivity and preventable defects due to communication difficulties, time-zone misalignment, misunderstood directions, etc.

Like defects and delays, these "hidden costs" (which are especially prevalent amongst offshore vendors) may seem insignificant at first—but often represent significant sums of unanticipated expenses. In fact, the hidden costs of offshore vendors have been estimated to increase the cost of their services by 65%![12]

To mitigate these wasteful expenses, Lighthouse Technologies, Inc. offers Rural Software Testing: an onshore alternative to offshore testing. By delivering higher quality, better performance, and less hassle, our onshore teams can drastically reduce hidden costs—which is why our prices match, and are often lower than, the total cost of offshore vendors.

[12]CIO Magazine, 2003

TAKEAWAYS AND FINAL THOUGHTS

It's easy for online retailers to get overwhelmed when trying to deal with the 3 Perils. While the sheer amount of options a company has to overcome these provides a high degree of flexibility, it's more common to stray into "paralysis-by-analysis" territory.

When that happens, keeping these four things in mind will help even the most vulnerable IT organization stay on course.

1 SOFTWARE PROJECT FAILURES ARE AVOIDABLE

Poor user experiences, declining customer loyalty, and project scale-backs/cancellations don't happen by accident. They're the inevitable outcome of countless cracks in the foundation—wrought by the 3 Perils of Software Development™: Defects, Delays, and Dollars.

By taking precautionary measures to overcome them, you can insulate yourself from their worst effects.

2 HAVE A PROFESSIONAL TESTING TEAM UTILIZING A PREDICTIVE METHODOLOGY

With all the trouble defects can cause, it's important to take them seriously. And while anyone can call themselves a software tester, only those with sufficient training and knowledge can ever call themselves professionals.

Once equipped with a metrics-based, predictive testing methodology (like Lighthouse's True North Testing Methodology™), a professional testing team should be the only people that you entrust your software to.

3 SUPPLEMENT YOUR TESTING EFFORT WITH AUTOMATION AND COST-EFFECTIVE ONSHORE PARTNERS

Whether they're in-house or outsourced, a professional testing team is vital in eliminating the effects of defects and delays in the SDLC.

Likewise, partnering with vendors like Lighthouse for test automation and Rural Software Testing services helps reduce budgetary strain by cutting testing budgets by 50% while mitigating the hidden costs of offshore vendors.

4 HONESTLY EVALUATE YOUR TEAM, IDENTIFYING WEAKNESSES AND LOW-HANGING FRUIT

Self-evaluations are never easy. Most of us have positivity bias—leading us to see things with rose-colored glasses. Unfortunately, when it comes to IT process improvement, this can be misleading—especially in an industry like Online Retail/eCommerce, where customer expectations are stringent and loyalty is fickle.

That's why an outside opinion is so valuable. A software testing/QA expert like Lighthouse Technologies, Inc. can produce an independent, metrics-based assessment of your organization that's free of bias.

Once the high-risk areas have been identified, you can start correcting the lowest-hanging fruit either on your own or with the partner.

In the end, overcoming the 3 Perils of Software Development™ comes down to two things: the knowledge to understand/identify the weaknesses in your own organization and the resources to fix them.

If you're finishing this white paper, you've already accomplished the former—the only thing that's still standing between your organization and the 3 Perils is the will (and budget) to overcome them.

Your customers have made their expectations clear. Show them you're committed to maintaining their loyalty by preventing the 3 Perils from marring their user experiences.

"TECHNOLOGY IS NOTHING. WHAT'S IMPORTANT IS THAT YOU HAVE A FAITH IN PEOPLE, THAT THEY'RE BASICALLY GOOD AND SMART, AND IF YOU GIVE THEM TOOLS, THEY'LL DO WONDERFUL THINGS WITH THEM."

– Steve Jobs

ABOUT THE AUTHOR

Senior Practice Manager at Lighthouse Technologies, Julie has over 20 years of industry experience and excels in helping software testing teams maximize their potential by tailoring their processes toward a metrics-driven, risk-based methodology.

ABOUT LIGHTHOUSE TECHNOLOGIES, INC.

A trusted partner of retailers worldwide, Lighthouse Technologies, Inc. offers a range of capabilities across three independent practices: Software Testing, Software Quality Assurance (QA), and Oracle EBS Consulting.

We see what others don't. No hunches, no best guesses, just data-driven metrics analyzed by our certified professionals—followed by comprehensive resolutions that we put into action. Regardless of the scale of your undertaking, our True North Testing Methodology™ exposes the dangers hidden beneath the surface of your projects, while our customized solutions ensure they deliver business value on time and on budget.

At Lighthouse, our success is measured in the revenue you earn, the hours you save, and the money you don't need to spend.

WHAT WE DO