Apsana, Inc.

Healthcare Testing

- EXECUTIVE SUMMARY
- NAME THAT PROBLEM
- WHO IS IMPACTED?
- CLASSIFICATION OF SYMPTOMS
- ROOT CAUSES

June 2013

CHALLENGES WITH CONTEMPORARY TESTING APPROACHES IN HEALTHCARE INSURANCE PROJECTS

Executive Summary

Most of software testing in the health insurance industry is beset with challenges. These challenges affect organizations undergoing large process or technology changes. These challenges can be classified into six major categories: *Planning*, *Knowledge Gap*, *Time Constraints*, *Insufficient Resources*, *Quality* and *Cost*.

These six major categories of challenges are themselves traceable to three root causes: *Organization Design*, *Need for a Dual Viewpoint* and the *Lack of a Testing Architecture.* As with most problems, the first and most important step to developing solutions, is to first accurately diagnose the problem. That is goal of this edition.

- Satish Nagarajan

Name That Problem!

If you have said, heard or experienced these problems then you might be using an approach to software testing that is not working:

- "Garbage In, Garbage Out"
- I am constantly firefighting- running from one crisis to the next.
- I can't get my business process owners to review and approve my documents in a timely manner
- How does the testing document relate to my process?
- I can't tell the value-add of testing.
- Testing costs too much!
- Testing always sand bags their requests for budget & schedule!
- PMO always cuts the testing budget and schedule!
- We never have enough time to complete all planned tests
- Testers don't know the business!
- Business does not have the time to train the testers!
- A large number of defects being rejected as "User/Tester Error"
- Creating testing dashboards is a full-time job
- Dealing with Project Change Requests is a full-time, high stress job!
- Test plans are obsolete as soon as they are written!
- I can't reuse any test cases!
- How can I tell when testing is complete?

WHO IS IMPACTED?

Many organizations experience various symptoms of these testing challenges. Impacted organizations include:

- ✓ Organizations undergoing large scale technology change like a new enterprise or departmental system implementation or an important upgrade to them
- ✓ Organizations implementing large scale process changes requiring significant changes to their system configuration and automation
- ✓ Organizations responding to new mandates, regulations or market conditions like Health Care Reform, BCBSA mandates, ICD-10, Health Insurance Exchanges and more

Impacted Roles: Testing Leadership, Test Analysts, IT leadership, Business Leadership, Business and IT SMEs.

Impacted Business Processes: Customer Service, Membership, Billing, Actuarial, Account and Finance, Claims, Medical Management, Network Management, Pricing, Government Relations, Legal and Compliance, Sales and Marketing.

CLASSIFICATION OF SYMPTOMS

In order to facilitate further analysis and arrive at the root causes it is helpful to categorize and classify the symptoms in to six groups:

- Planning
- Knowledge Gap
- Time constraints
- Incorrect resource Levels
- Quality
- Cost

Planning

- Plans are written in highly technical language and are not easily understood or reviewed by business partners.
- Plans are point in time documents that are not updated as the project changes due to change requests or unplanned events so either the plan becomes obsolete or a lot of effort must be made to keep the plan updated.
- Plans do not assist with evaluating change requests and other project changes as they occur
- Even though changes should be expected and planned for, change requests are handled on an exception basis by Testing Leadership.
- Plans do not specify planned test coverage up front so it not possible to verify upon completion of testing if the plans were accurate
- Plans are difficult to justify as the traceability to project scope and requirements is not clear and reviewable
- Plans are not clearly tied to Business Processes and Parameters so business owners don't understand what is being presented.

***Result**: Plans are written to satisfy an SDLC checklist but are not very useful later in the project. Plans are difficult to review and justify so get only proforma approvals. Since planned coverage is not clearly delineated it is not possible to verify actual coverage upon completion of testing.*

Knowledge Gap

The Knowledge Gap manifests itself in a number of ways:

- Testers don't understand the business and are unable to detect implied and undocumented assumptions
- Testers don't have enough background knowledge to elicit critical business conditions from interviews with business SMEs
- Testers are constantly asking for help from business SMEs or require extensive training in the business process
- A significant number of defects are classified as tester error
- Test cases are superficial and are not able to validate nuances of the system functionality and business processes
- Test cases have different levels of depth across different domains according to the knowledge levels of the individuals
- There are often one or two critical resources who serve as testing SMEs for each domain who become bottleneck resources. They are also hard to replace in case of unavailability.
- There is a wide range of productivity across team members even accounting for individual characteristics because of the range of systems and business knowledge.
- Ramp-up times for new teams and new team members is large and productivity increases slowly

***Result**: Assembling an effective knowledgeable team can be a daunting task. Testing team's reputation suffers in the meantime. A lot of testing has to be repeated, increasing costs and increasing project schedules.*

TESTING CHALLENGES

Time Constraints

- Most projects have aggressive schedules. Testing being at the end of the end of the SDLC is impacted by any schedule slips earlier in the SDLC, i.e. Testing is expected to make-up for schedule slippage to keep the overall project on schedule.
- Since testing is dependent on and consumes deliverables generated in earlier stages of the SDLC, delays or changes can have large impacts on the testing schedule. A late deliverable or a late change to a non-testing deliverable can impact a number of testing deliverables. This can create a cascade of schedule slippages in testing.
- "Crashing the schedule" requires finding additional resources or taking process short cuts or eliminating certain test cases altogether.
- Just determining the impact of change requests can become a high overhead task for the Testing Leadership team. Therefore impact analysis is not done and cost benefit analysis is not measured. Incapable of doing what if scenarios to determine the best changes to accept and those to reject.
- Many current Test Plans do not anticipate these types of situations and so quickly become obsolete in these situations. Further the Test Plans do not provide any guidance on measuring the impact of these situations to Testing and to the overall project.
- A significant portion of the time and effort spent by many Testing organization Leaders is in dealing with this situation.

***Result**: Aggressive schedules are normal for most contemporary large projects. The test process is routinely under schedule stress which can compromise budget, productivity, effectiveness and quality unless dealt with proactively.*

Incorrect Resource Levels

- Due to schedule constraints, the knowledge gap and poor planning many testing organizations discover they have insufficient highly skilled workers in one or more areas late in the schedule.
- The resource constraint can arise due to unplanned change requests, upstream delay unexpected resource availability issues (family emergency, resignation, illness etc.) or just poor planning.
- Change requests and requests to crash the schedule can exacerbate tight resource budgets and acquiring additional knowledgeable resources can be challenging
- Proactive test managers have sometime resorted to over staffing during certain phases to retain the flexibility to deal with the unplanned but expected schedule crunch later in the test phase.

***Result**: Either teams are overstaffed for most of the project so that they can have enough staffing during the crunch or they are understaffed at the crunch. Both approaches result in additional costs and can have quality implications*

Quality Challenges in Testing

Many testing programs define Quality as "Conformance to Requirements". Many projects have no formal documented requirements. If documented requirements exist they may be based on undocumented assumptions regarding "General Process Knowledge". Many legacy systems do not have current documented requirements and have grown over time. If test cases are to be based on these requirements then this creates a particular quality challenge.

Often test cases are not clearly matched to business requirements early in the test development process so it is hard for others to determine what is being tested and why. This so called Requirements Traceability Matrix is at times only developed upon the completion of the test phase as a post-execution audit. This makes it hard for Project Leadership and Business Owners to understand the Quality aspects of the solution and build confidence in the system delivery process

Testing can traditionally tie defects to particular system components under test however they have a particular challenge tying defects to business processes or scope components. This makes it difficult when a project is facing a difficult "Go/No Go" decision because it is not always possible to translate Defect Reports and Test Execution Metrics into business process readiness.

***Result**: Testing is often quite successful identifying defects when the test case matches the requirement but can miss the mark when the requirement is unknown to the testers. Testing metrics also do not provide a ready guide to the status and readiness of the system for business use.*

PRIMARY QUALITY GOALS

Testing

Testing processes have three primary quality goals:

1. Identify defects in build (infrastructure, configuration and programs)
2. Increase business user confidence in the system
3. Provide guidance to the other SDLC teams as to quality improvements

COST

Direct Cost & Indirect Cost

There are multiple cost dimensions that need to be looked at when evaluating the cost of Testing:

Direct Costs

- Unit Cost
- Total Project Cost
- Total Lifecycle Cost

Indirect Costs

- Cost of Quality
- Project Cost Avoidance

Apsana , Inc.

Most organizations are only able to measure a portion of the Direct Costs. Unit Costs are usually easy to determine from Payroll costs and from Purchasing when using contract labor. Total Project Costs are more difficult to determine because of challenges with cost allocation of part-time SMEs to the testing process but for larger projects and programs the measured values are not representative of the true costs…

There is typically little to no measurement of the other costs including: Total Lifecycle Cost and Indirect Costs. Total Lifecycle Cost measures the cost of maintaining the test team and the test artifacts (tools, documents, repositories, test cases etc.) between projects. These are usually categorized as operating costs across multiple budgets.

Indirect Costs are by some estimates likely to be much larger than the Direct Costs and have two components. Cost of Quality is all avoidable costs associated with remediation and rework of a defect that could have been prevented or detected and corrected early in the Software Development Life Cycle. These include defects found in later testing cycles and in production. There are varying estimates for the Cost of Quality ranging from 1X to 10X and more of the cost in the project.

Project Cost Avoidance is directly measurable but not often measured. Most large projects have a stable "burn rate" that can be estimated on a weekly or monthly basis. This is the amount of money the project spends on direct costs for each week or month it exists. Since Testing is always in the Critical Path of a project's schedule any reduction of the critical path schedule results in saving to the project. For large programs the "Burn Rate" of the project for a couple of weeks could easily represent the entire testing budget. Put another way if Testing could shrink the critical path of the project by a couple weeks then it could be completely free.

Being effective (reducing the Cost of Quality) and efficient (increasing Project Cost Avoidance) within Testing is far more important than managing Direct Costs. However many organizations have strong imperatives to manage Unit Costs and not manage all these other costs enacting the proverb "Penny wise; Pound foolish."

Supporting Routing Releases	Supporting Large Programs & Projects
Requires a small multi-disciplinary team that can support a variety of business and technical domains.	*Requires a large multi-focus team. Each sub-team will require deep expertise in particular set of domains*
Testing is rapid and informal. SDLC might be shortened with fewer formal checkpoints.	*Testing should follow a formal SDLC with standards and formal checkpoints*

Organization Model

Testing as an ongoing support function and testing for large projects are entirely different animals. Many organizations use a scaled version of the same model to support both kinds of testing needs which can cause a number of problems.

If a "Routine Release" team model is scaled to support a large program then it is likely the team will underestimate the resource requirements, the knowledge requirements and compliance requirements. The core team members will execute according to the normal practice which will fall short of the formal SDLC standards resulting significant "culture shock". A lot of testing leadership time will be spent in unplanned education, training and compliance activities. The team will not have the depth of expertise required initially resulting in a lot of friction with business and development teams.

If the "Large Program" team model is scaled down to support routine releases then the time will likely be too large, too slow and too formal. The team will have challenge working from informal documents and hand-offs. A lot of testing leadership time will be spent bridging the methodology gap between the informal SDLC being followed by other teams and the formal SDLC expected by the testing team.

Testing architecture

IT teams typically use several standard documents including architecture diagrams, functional and technical specification templates and coding standards. These provide a lot of guidance and structure to the team and facilitate cross team communication.

Large business processes similarly have clearly defined training manuals, role definitions, desktop procedures, workflow diagrams and operational reports that allow them to manage their daily workload. Testing sits at the nexus of these two domains and has to bridge these two different views. If the test team does not have a usable and enforced architecture including standards, templates, procedures, workflow, roles and tools then the productivity and quality suffer.

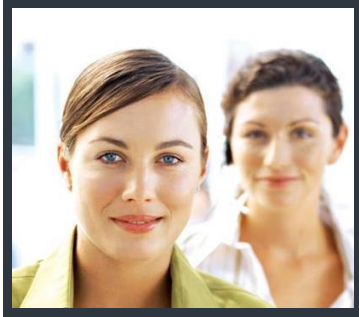
Most established test teams have documented methodology and some standards and templates. Many organizations have also invested in testing tools. However the big problem is there is at most a superficial compliance to these standards. Also the tools, templates and standards are not tightly integrated across the dual viewpoints – business and IT – making using the templates more time consuming.

The challenges that arise from this situation include inability to reuse past deliverables. It is also very difficult to improve the accuracy of budget and schedule estimates across projects. The testing costs do not go down with experience, in fact might actually rise over time.

ROOT CAUSES

Careful analysis of these testing challenges across a large of number of projects and large number of organizations has allowed us to determine the following three root causes:

1. Organization model
2. Testing Architecture
3. Dual view – Business Process and Systems



DUAL VIEW

Business Process and Systems View

Dual View Points

Most test teams are hosted and staffed by the IT department. As such their native viewpoint is a systems viewpoint. Even if the team is considered a "Business Testing Team" it is likely staffed with folks who have an IT background. This is very useful when using a technical viewpoint to testing but the business process viewpoint is underappreciated. This can account for a number of testing challenges including:

- Inability to understand assumed and undocumented requirements. Often requirements are documented as "deltas" to current behavior. Current behavior is often poorly or not documented and is assumed that everyone understands them.
- Requirements can appear contradictory but are not because they apply to unstated contexts.
- Test plans and test cases are written to technical specifications which are usually easier for IT people to understand however they are harder for the Business Process owners to review and approve.
- Testing budgets and schedules are derived based on estimated IT metrics – feature points, function points, lines of code, number of components etc. However when these estimates are developed (during planning) there is a large error margin in these estimates. Further it is hard to connect these estimates to high level scope which is defined in business terms.
- A lot of complexity and testing effort is derived from business process complexity. One technical component may be used in multiple process contexts and may map to multiple requirements. There is a significant many-to-many mapping relationship between requirements and technical components and using one view over another causes a significant error.
- Testing serves three large stakeholder communities: Business, IT and Project Management. Each stakeholder community has a different set of needs and not considering them organically throughout the testing lifecycle can cause significant overheads.

The challenges that arise from this situation include increased Testing leadership overhead creating status reports and dashboards to satisfy different stakeholder groups. A lot of time has to be spent resolving miscommunication and misunderstanding. The stakeholders have a poor appreciation of the testing process and its value-add. A lot of effort is required to justify and defend testing budget and schedule requests. Testing is often pressured and surprises the Project Management Office with change requests. It is hard to provide real-time traceability of test results and defects across both systems and business process domains.

If these challenges apply to your projects, please contact us. We can help you.

Satish Nagarajan is a Principal with Apsana, Inc. a management and IT consulting firm. He has over 20 years of experience in health insurance System Implementations. His clients include some of the largest health insurance companies in the US.

Mr. Nagarajan has a MS in Computer Science and a MBA from the Kellogg Graduate School of Management.

About the Author

Apsana
evolve your business

214 302 7286
www.apsana.com
consulting@apsana.com