Quality Management

Defensive Project Management

Elite Partner
Software

Northway Solutions Group
9005 Overlook Blvd • Brentwood, TN. 37027 • Ph: 866.611.8762 • Fax: 866.611.8762
www.northwaysolutions.com

Are you tired yet of getting your testing schedule collapsed to meet a release date that was committed to before the project even started?

Every day I hear from frustrated QA managers who have just been informed by project management that their six-week testing schedule has been reduced to two weeks or less. It usually involves some sob story about how the development team is a month late on delivering their code because of changes requested at the last minute by the "customer" (isn't it always someone else's fault?). In any event, the testing team is expected to suck up the lost time and get all of their testing done in a mere fraction of the original schedule because "we have to make the date".

You may not like the response I have for these QA managers. I typically ask them, "What have you done to prevent this from happening?" This is when the manager usually explodes from sheer pressure, and dumps a laundry list of reasons for the project's poor condition: Management committed to a date before the project even kicked off. There are problems with the requirements; either they are non-existent, vague, incomplete, or not well documented. There are problems with the involvement of quality resources, such as being left out of key meetings. There are problems with the development team producing unit-tested code. There are problems with ever-changing requirements, and a lack of any concern for managing the rate or scope of the changes. There are problems getting enough test resources to adequately test the functionality. And finally, there are problems with management support for testing, test environments, or managing defects. Now that the QA manager has had the opportunity to dump all of the pent up frustration, I can once again ask the question, "What have you done to prevent this from happening?" The now pressure-relieved manager typically responds that they provided a schedule up front, and they made sure to remind the PM at every opportunity that they were not going to make the date. It is easy to play the role of a victim. It relieves us of our

> It is easy to play the role of a victim. It relieves us of our responsibility to manage and take accountability for the condition of our projects.
>
> It is also intellectually dishonest.

responsibility to manage and take accountability for the condition of our projects. It is also intellectually dishonest.

This is a typical project scenario for many organizations. Finding an organization where quality is an inherit part of the methodology, and where everyone takes ownership for building quality into the process and products is a rare find indeed. For the rest of us, we have to live in that chaotic world where we are doing our best to "test" quality into the end product. So if this is the environment that most of us work in, we have to find better ways to manage the chaos of that environment to give ourselves the best possible chance for success. The first way to get yourself out of the hole you are in is to stop digging. We will now explore some strategies for better managing the quality process within our organizations, and give you some tools that you can begin to use tomorrow to start to lift yourself out of the hole you are in. Doing the same things you always have done, will get you the same results you have always gotten. It is time to make a change. Let's explore just four strategies that will help you to become successful on your projects:

- Begin defensive project management
- Establish the fundamental value of quality
- Solicit assistance from leadership
- Identify the project degrees of freedom

The balance of this article will take a deeper look at each of these strategies and how you, the QA manager, can use them to change behaviors and outcomes in your organization. Imagine a world where you have the time you need to fully test and verify that the product meets its intended need. This is not the unattainable goal!

Begin Defensive Project Management

If I were to ask each of you to tell me who was responsible for quality, I am pretty confident that most of you would say, "Everyone is responsible for quality", and you would be absolutely correct. On the other hand, if I were to ask each of you who is responsible for project management, most would likely say the Project Manager was responsible. Nothing could be further from the truth. Just as everyone is responsible for quality, everyone is responsible for managing the project. The person identified on the project contact list as the Project Manager can't possibly do everything it takes to successfully manage a project. As the QA Manager, you have the responsibility to manage the testing aspects of the project, not the PM.

Here is where I will take it a bit further. Not only does the QA Manager have a responsibility to manage the testing aspects of the project, they have the responsibility to defensively manage all of the aspects of the project that could potentially impact testing. "What? As a QA manager, I have to manage the entire project?" If it impacts or could potentially impact testing, my answer to you is "absolutely"! If you are tired of being the victim of the bad behaviors of other teams or functions, then you have to help manage their delivery. I have witnessed too many projects, where the development team is allowed to miss dates, and functionality, and unit testing, and code freeze deadlines, only to have QA suck it up. If you want to be successful in getting the time you need to adequately test, then you need to ensure that all teams are managed to their committed dates, and deliveries.

If you have been in software development for any amount of time, you have likely heard the term "defensive programming". This term is used to describe a programming design technique where the developer designs their program to be very robust. It uses the adage: "*trust nothing you receive…validate everything you send*". This strategy has the programmer coding their application to be suspect of anything they receive as an input, and doing checks up front to validate the input received is good, making for better more stable programs. Defensive programming places the power in the hands of the developer, as they shield themselves from the bad habits of others. The same adage holds true for project management. You as the QA manager have the ability to better shield yourself from the bad habits of others, by taking a proactive role in managing the management of those processes, teams, and artifacts that drive the success of the quality team. While the Project Manager has accountability for the day-to-day management of the project, you can have an enormous impact by watching the watcher, and helping the PM to manage the activities of those teams.

One of the biggest issues that I have with Project Managers is that they don't manage to interim project milestones in the same passion and urgency with which they manage the end-date. As the QA manager, you have the responsibility to ensure that they recognize that the project isn't red when the QA team is late - it's red when any interim milestone is late that could impact the end-date. You need to ensure that the PM is managing the Business Analysts and Developers in the same rigor with which they manage the QA team. When was the last time you heard a PM asking the BA team if they were working nights and weekends to make their requirements baseline date? You probably have never heard of such a thing because they always assume they can make up the time later, and you know who always pays when they don't make up the time. When it comes down to the wire, the QA team will be the one giving up holidays, weekends, and working extended hours, while the developers and BA teams get a much needed break. If you are not making sure your PM is managing those interim dates, don't be upset when you are forced into the corner.

Remember, Project Managers often report directly to the application manager who made the commitment to the customer in the first place. They often see themselves as having to make a decision about keeping their management happy and doing what is best for the customer. They may not understand that they place the entire project in jeopardy when they cut quality time in the schedule. They mistakenly believe they are serving the customer by making the date at all cost. If you are not going to every project status meeting then make sure you are invited, and under all circumstances attend. Do your homework first. Look at the tasks to be completed, and question the ability of each team to deliver on schedule. If it looks as though one of the teams isn't going to make their delivery, then ask what they are going to do to get back on schedule. If the scheduled delivery date is going to be missed ask why, and identify what is going to be impacted by them being late. Ensure that all down-stream impacts are addressed immediately, and that the project status correctly displays the downstream impact. If requirements are late, make sure the project is identified as red, unless an acceptable mitigation plan is in place. No one is going to look out for your team, but you, so don't rely on the PM to do it for you.

Establish the Fundamental Value of Quality

When I hear about testing cycles being significantly reduced on virtually every project, what I am hearing is that there are one of two issues with the team or organization. The first issue may be that the organization doesn't fundamentally believe that the testing schedule is accurate or true. As an example, this typically occurs when a team originally scheduled six weeks for testing, but was forced to reduce the schedule to two weeks and still deliver the agreed upon testing scope, or so it would appear. In reality, the team likely didn't complete all of the testing, and only did a cursory job of testing the most critical features, resulting in latent defects in production causing higher support costs, and less confidence in the testing department. This turns into a vicious circle as the testing team begins to pad schedules with the knowledge that they are going to be asked to reduce it, further adding to the lack of confidence by leadership in the schedule.

The only way to combat this perception is with metrics. The numbers have to be absolutely accurate, and any changes to them have to be adequately documented and justified. Use a standardized estimating model to establish your quality project plan. Whether the model is based on function points, or on some other estimating method, make sure that the model used is consistent. Have a boilerplate schedule that identifies the entire task list. Create a model of test case design effort, test execution effort, defect management effort, etc. This model will then provide you a better ability to justify the effort you are identifying. Always remember to go back after each project to refine the model based on actual effort expended.

If you reduce the schedule by working overtime and weekends, make sure that you document that the schedule wasn't reduced, only the calendar delivery was reduced by increasing the work hours and days worked. Ensure that everyone understands that this is not an acceptable long-term trend for the project team, as you will have a higher than normal turnover. Everyone can get behind an emergency need to work overtime, but you will begin to loose resources if every delivery is an emergency.

Lastly, manage the quality effort throughout the lifecycle and that you are constantly reporting on status to leadership. I highly recommend using a test management solution, such as the HP Application Lifecycle Management™ (Quality Center®) solution to manage this activity. These toolsets provide you with very robust reporting capability. It is hard to argue with actual numbers generated from these toolsets, and they help the entire team to make educated decisions about testing efforts. When schedules are reduced and testing is sacrificed, make sure to produce post-production defect metrics that capture the cost of poor quality. If you want the organization to change, there has to be a problem to solve. Production issues always get top attention.

The second issue causing reduced testing cycles is that the organization doesn't fundamentally believe in the value of the testing team. There can be many reasons for this but commonly there is a lack of confidence in the caliber of testing that is accomplished as demonstrated through post-production defects. This lack of confidence can be due to the testing team taking a more ad hoc approach to testing, or the fact that testing is very superficial and doesn't discover enough critical defects for the amount of effort expended. The relative value of the testing to the effort expended is in question. These teams usually rely heavily on the customer to identify defects during User Acceptance Testing (UAT). This is an organizational reputation issue.

To repair or improve the reputation of the testing organization is a much harder accomplishment than to produce accurate schedules and execution metrics. First and foremost, testing requires a skilled professional. It is not the department where all the developers who couldn't cut it as developers should be placed. Nor is it a career step toward becoming a developer, project manager, or business analyst. It is a discipline that requires a strong education in quality as well as, business, and technical prowess. If you have a QA team made up of the dregs of your organization, it is probably time to retool your QA function. If you want bad testing, hire anybody who will do the job. If you want superior testing, you must seek out true testing professionals, or invest in the best and brightest your organization has to offer to develop them from within.

Establish a risk-based approach to testing. This means that all functionality is prioritized and risk rated, and that all testing is designed to accommodate the relative risk and priority of the functionality being tested. In this environment the highest priority and highest risk features and functionality gets tested first, and with the most rigor. In the event schedules must be compressed, the team can be confident that the most important features are thoroughly tested prior to launch of the application. Make sure that nominal testing is segregated from challenge testing. This means that nominal or positive tests that verify that *the system does everything it is supposed to do*, are separated from challenge or negative tests that verify *the system does nothing it is not supposed to do*. When these tests are separated, regression becomes much quicker and easier as challenge can be eliminated in regression rounds of testing where changes to that functionality have not occurred. All too often massive test cases with nominal and challenge mixed together are run for each regression causing inefficiency and wasted effort. Ensure that challenge is absolutely completed for all critical features of the application as identified by your prioritization. Always produce metrics that identify the number of features tested and the severity of defects discovered by your testing team. When the development team recognizes that you are finding the defects rather than the customer, they will become your strong ally.

Solicit Assistance from Leadership

One of the biggest challenges that QA managers identify is pressure from their direct management to meet a date, usually veiled in terms of customer demands. The conversation usually begins with the manager telling the QA manager that the customer is demanding the solution right now and that the IT organization must do everything possible to deliver it on time, including compressing testing schedules.

> If you ask any customer what they mean when they want it "right now", they mean "correct now" not "bad now".

First of all, I believe that if you ask any customer what they mean when they want it "right now", they mean "correct now" not "bad now". With that being said, most of our customers would rather we do what it takes to develop a good product, but want to understand why it will take longer than they anticipate it should take. The problem is that IT has traditionally over-committed and under-delivered so the customer has a low level of confidence that IT will deliver a quality product in the first delivery. So, the Customer holds IT to tight timelines realizing it will take several iterations to get to a final product that satisfies their business need. We as an IT organization have to change that perception with our customers. We can only do that by delivering on commitments. The last section of this article addresses how we begin to accomplish this transformation of our well-earned reputation.

So how can you get your manager to help? In many IT organizations managers are rewarded based on what they deliver, not how. I have attended project celebrations where senior management was walking around patting themselves on the back for delivering a product to production that was millions of dollars over budget, with major defects, and only a portion of the functionality originally prescribed. As a matter of fact the only reason the release date was met was because it was changed multiple times. We celebrate getting something; anything delivered to production on the release date.

So how do we change this “deliver at all cost” mentality? You have to solicit your management’s support for delivering it “correct now”. Most managers want to help. You really stroke their ego when you honestly and sincerely ask for their help. Managers are wired to be problem solvers, and will go out of their way to address a problem if you ask for help. The problem is that most of us don’t want to ask for help because we see it as a sign of weakness, when in reality it is a sign of strength. When was the last time you went to your manager and informed him or her that you really needed their help to be successful? In his book “*The Servant – A Simple Story About the True Essence of Leadership*”, James Hunter describes how the role of a manager is to serve their employees by ensuring that they have everything they need (not want) to be successful and accomplish their goals. It is not your manager’s job be the barrier to your success but to remove barriers. Be honest with your manager and ask them to help you strategize about how to get the things your team needs to be successful. Take special care to look at how you are serving your team at the same time. Are you removing barriers from your team being successful? If you are not managing effectively barriers to your team’s success then you are not serving your team. Is asking them to work just one more weekend, just one more long night really enabling them to be successful as a team?

Identify the Project Degrees of Freedom

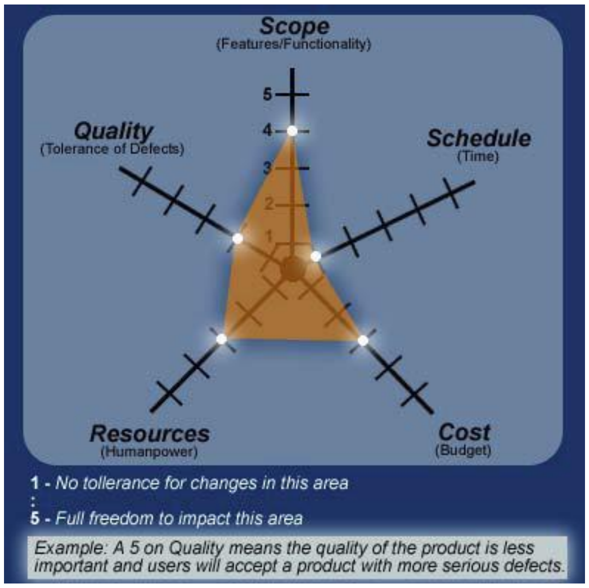
Conversations about compressing the testing schedule never happen at the beginning of the project, they always happen in the heat of the moment, usually in a challenging situation when it's too late to have a calm and analytical discussion. Almost always, the conversation is about an emergency situation. Development is late, and we only have a couple of weeks and sometimes days to get make the schedule. It's too late to add resources; it's too late to get consultants; it's too late to reduce functionality; and, it's too late to change the schedule because the customer has already scheduled training and the marketing blitz. So what gives? Quality of course! We put all of the pressure on the over-worked, understaffed testing team and expect them to once again "save the day!"

It doesn't have to be this way. Imagine a project where a discussion was held at the beginning of the project between IT and the customer, in which, all of the aspects of a project were negotiated up front when everyone was calm, and good business decisions could be made. What a difference it would make to know up front that you can adjust the amount of resources on the project to make it successful, or that schedule could be extended to obtain the desired functionality. No more would you be faced with making a choice between two bad choices! You can have it all!

In his book "*Creating a Software Engineering Culture*", Dr. Karl Wiegers identified the five dimensions of a software project as:

- **Scope** – Ability to adjust the features/functions delivered in the project
- **Schedule** – Ability to adjust the delivery schedule of the project
- **Cost** – Ability to adjust the budget/cost of the project
- **Resources** - Ability to adjust internal resources on the project

- **Quality** - Ability to adjust the quality expectations on the project (number of acceptable defects)


Scope
(Features/Functionality)
Quality
(Tolerance of Defects)
Schedule
(Time)
Resources
(Humanpower)
Cost
(Budget)
1
2
3
4
5
1 - No tollerance for changes in this area
5 - Full freedom to impact this area
Example: A 5 on Quality means the quality of the product is less important and users will accept a product with more serious defects.

Degrees of Freedom Graph

Dr. Wiegers identified that for each of these dimensions on the project, there is a level of freedom that each one of these dimensions can be adjusted on a given project. This level of freedom is called a *degree of freedom*. Each degree of freedom has an acceptable range of freedom from 1, meaning no ability to adjust, to 5, meaning a great amount of ability to adjust. Below you will find an example of a graph that displays the five degrees of freedom for a given project. The graph at the right shows how the degrees of freedom are displayed for a given project. In the example on the right, Schedule is identified as a value of 1 - indicating the agreed upon schedule is a primary driver of the project which cannot be changed. However *Scope* might be a 4 - indicating that features and functionality may be removed in order to meet the more important schedule constraint. Each organization would identify the criteria for adjustment along each of the five dimensions of the project.

The key to the success of this graph is to obtain agreement between the Customer and IT *at the initiation of a project* to identify the degrees of freedom for each of these dimensions. This graph should be a required component of the project chartering process, and both IT and the Customer should "sign off" on this graph. If this graph is an approved part of the project charter, the implementation team no longer has to have heated debates over what to do in the case of an impacting event in the project. The team can quickly refer to the charter to identify where to make appropriate project adjustments. If quality and schedule have low degrees of freedom, while scope and cost have higher degrees of freedom, the team can make the appropriate choices to add budget or reduce features in order to meet the more critical release date.

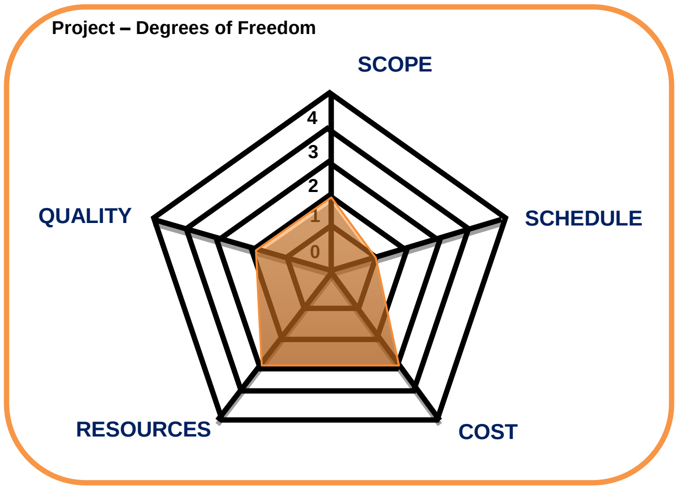
This contract between IT and the Customer removes pressure from individual project team members and manages the project to a predefined standard, removing personalities and politics from the decision process. It protects the much maligned quality team, and ensures that either quality is given sufficient recognition and support, or that quality is relieved from burden as defects are accepted in advance by the Customer. This is truly a "win – win" outcome for both the Customer and IT, and helps to ensure that more projects are delivered on time, within budget and at an acceptable quality level.

Each organization will need to determine what components make up each of the degrees for each dimension. Below is an example of a spreadsheet that implements automated capture of each dimension. As you can see the user selects a degree from each of the dimensions, which then updates the graph automatically. Both the graph and the selection criteria can be imbedded into the charter document. Both the Customer and IT then signs off the charter document making this negotiated decision process an integral part of the project delivery.

SCOPE	
○	All features/functionalities are required to be delivered in this release without exception
[✓]	All high & medium priority features/functionalities are required to be delivered in this release
○	All high priority features/functionality are required to be delivered in this release
○	High priority features/functionalities may be removed to meet project objectives with customer approval
○	There are no requirements to deliver specific features/functionalities this release
SCHEDULE	
[✓]	The identified release date must be met at all costs for this project
○	The release date is critical and deviations from it should be approved by the customer
○	Every attempt should be made to meet the schedule; however, not at the expense of the other critical areas
○	Regular adjustments to the schedule are acceptable at each of the project phase tollgates
○	The delivery of the requested functionality at the appropriate quality level should drive the release date
COST	
○	Budget is absolutely fixed for this project. When budget runs out the project is cancelled
○	Budget is fixed. Senior leadership must approve continuing project for budget additions
[✓]	Budget may be added only to reach the quality or schedule goals. Scope changes require new budget
○	Budget may be added to this project to meet the needs of the project (e.g. adding consultants)
○	This program has been identified as a critical project without budget limitations

RESOURCES	
○	The number of internal resources is fixed and no additional headcount can be moved to this project
○	Headcount may be moved to provide additional assistance only if moved from lower priority projects
◉	Headcount may be moved to this project at the discretion of IT leadership
○	Headcount may be redirected to this project, including offshore partners at the discretion of the team leader
○	Headcount may be freely moved to this project to get it accomplished
QUALITY	
○	The product must meet all functional and performance criteria without known defects
◉	No severe defects allowed. Only serious & minor defects allowed with approved work around
○	Serious defects in low priority features only. An approved work around is required
○	The customer is willing to review and sign off on any defects, with an approved work around
○	Defects in the end product of any severity are acceptable

Imagine the power in your hands as a QA manager, when the PM approaches you to reduce the testing effort. You can pull out the charter and begin to have an educated discussion about the ability to adjust project quality. You can then explore all of the alternatives based on a predisposed and agreed upon decision criterion. What a different world that would be!



In summary, QA managers are not powerless, and destined to be victims of the schedule, and bad project discipline. QA managers have it fully within their power to be champions for good project management taking ownership for the success of their teams, and ultimately the success of the products that IT delivers to customers. This article has explored four distinct strategies that QA managers can begin to employ tomorrow in their journey to great projects.

> ***Begin defensive project management*** – You are only the victim of others if you allow yourself to become a victim. Beginning tomorrow, take ownership of managing not only your tasks but also any tasks, which may impact your delivery.

Establish the fundamental value of quality – Make sure your organization fully understands the magnitude of testing efforts and that the organization values the benefit that testing brings to the process. Beginning tomorrow, commit to using repeatable estimating models that are updated after each project and that you produce metrics of where you are and the cost of poor quality.

Solicit assistance from leadership – Your manager should be your best ally. Routinely seeking his or her help in the delivery of your work, but also remember that you too have an obligation to provide your team with what they need to be successful. Beginning tomorrow, spend time with your manager soliciting their help in removing barriers to the success of quality.

Identify the project degrees of freedom – Understanding what can be manipulated within a project to ultimately deliver a successful project is key to success. Creating a project degrees of freedom graph and negotiating the level of freedom in each project dimension will remove pressure from the team and deliver higher customer and management satisfaction. Beginning tomorrow, create a graph for each project you and your team is working on. Work with your project manager and customer to gain agreement on the graph. Obtain agreement and documented "sign-off" of the graph. Work to implement the graph as a normal part of your project chartering process. Then relax and step back from fighting the fires!

While there is no magic formula for a project's success, utilizing the concepts and tools identified in this article will help you to become a more effective manager, and will help your organization to effectively deliver on projects. You will have an organization where testers are valued, and turnover is reduced. Ultimately your customer will gain confidence in the ability of IT to deliver on commitments and quality. Imagine that world, and then tomorrow, begin to make it happen.

About the Author

Brian Copeland is currently the QA Practice Director for Northway Solutions Group. With nearly 30 years of senior level experience in the software development industry specializing in organizational transformation and development, Brian has been instrumental in the testing of critical business systems, from mission critical applications to commercial software.

References

James C. Hunter. 1998. The Servant – A Simple Story About the True Essence of Leadership. New York: Crown Business Publishing

Karl Wiegers, PhD. 1996. Creating a Software Engineering Culture. New York: Dorset House Publishing

About Northway Solutions Group

Northway Solutions Group is a HP Elite Solutions Partner and Reseller for HP Software products Northway employs certified consultants with real-world experience who provide long-term solutions to the toughest business challenges. This includes providing training and implementation service of HP Software products

Contact

9005 Overlook Blvd

Brentwood, TN 37027 USA

Phone: 866.611.8762

Web: www.northwaysolutions.com

Email: Info@northwaysolutions.com