The Software Quality Advisor Online

Rice Consulting Services, Inc.
P.O. Box 891284
Oklahoma City, OK 73189
405-793-7449
405-793-7454 FAX
December 2001

Dealing with the Intimidation Factor in Software Testing

by Randy Rice, CQA, CSTE

Intimidation is something that everyone has experienced or given at some time or other, sometimes as the intimidator and other times as the object of the intimidation. Although intimidation is part of most project cultures, software testing is an activity that seems especially prone to experience the effects of intimidation as the giver, the receiver, or both.

I often tell beginning testers that everyone can be their friend throughout the test planning process, and even into the test execution process. When the first defect is reported, then the picture often changes. The task of defect resolution is often when defensive posturing emerges and can be seen as discounting defects as "features," "user errors," "anomalies," or even hallucinations on the part of the tester. Many times the defensive behavior degrades into blaming others. Testers are even blamed for the defects they find, which is like blaming the fruit inspector for bad fruit.

However, intimidation can go both ways in the software development process. Testers can feel intimidated by the knowledge and power of developers, while developers can feel intimidated by the critical review and power of testers.

While we will never be totally free of intimidation and its effects, I believe it is possible to be aware of the intimidation factor and work to minimize its negative impact on the project. In this article, we will explore the nature of intimidation, how it impacts software testing, and some definite ways to deal with intimidation.

What is Intimidation?

Intimidation is the exerting of power or force over someone to make them behave in a certain way. Police officers wear certain types of uniforms to project an image of power. Think about it. Which is more intimidating the "Smokey bear" hat clad state trooper or the bicycle cop wearing a helmet and shorts?

In the martial arts, sparring opponents will often come out with a loud yell, which is primarily intended to intimidate and distract their opponent. In baseball, the catcher will say all kinds of things about the batter's wife and mother to intimidate and distract the hitter.

From these examples we can see that intimidation is often an adversarial position between two or more people. Sometimes, the motivation to intimidate may be to achieve a common goal, such as a football coach intimidating the players in order to win a game. However, even in the context of a common goal, intimidation is a weak method to motivate using power plays.

When you take the idea of intimidation and adversarial stances into a project situation, it is no wonder that so many projects have difficulties. Instead of bringing focus and cooperation in the context of agreements and well-defined working agreements, people resort to the first technique they ever learned – intimidation. That explains why some projects resemble public school cultures. You can pick the grade level.

It is interesting that each of us has the ultimate control over intimidation in our lives. We feel intimidated when we give people power to make us feel a certain way. As long as we feel fear from an intimidator, he or she will continue to have a level of control over us. Once we decide that the intimidator has no control over us except to manipulate emotions, we can move past the fear and go on with out job.

An intimidator can change by learning how to deal with people in a disarming way. Many times people do not even know they are intimidating others until someone calls it to their attention. After coming to the awareness that one is an intimidator then it is possible to change. However, change is one of those things that requires continual attention and often the help and understanding of family and co-workers.

In other cases, the person feeling intimidation might be feeling that way because of self-imposed fears. This is especially true of people who create things and are reluctant to reveal their work to others. This explains why software developers might feel intimidated when releasing their work to an independent tester. If the feedback from the tester is constructive and focused on the product in a positive way, the perceived intimidation level can be reduced and even eliminated. However, if the feedback is critical, then the software developer will likely build a series of emotional defense mechanisms, which can range from disengaging emotionally from the project to anger.

How Intimidation Can Impact a Project

While the focus of this article is on dealing with the intimidation factor in software testing, most people with experience on software projects could relate plenty of situations where intimidation was used between customers, users, senior management, development staffs, documentation staffs, and groups other than QA and testing. The impact of intimidation exerted by a customer or user on the development staff can certainly cause major project problems. For example, the customer that insists on a software release in three weeks "or else," has single-handedly introduced perhaps the most significant project risk – an accelerated deadline. In this example, the intimidation often ripples through the development team, testers, documentation writers, trainers, customer support staff and everyone else in the software release process, including users.

Realizing that intimidation can be seen anywhere and anytime on a project, including software testing, for this discussion I will suggest the following major points of impact from intimidation:

Creating an Adversarial Culture

One of the first things that intimidation does is to build walls between people. These walls divide people into camps that seek to protect their respective territories and interests. This kind of division leads to the "us vs. them" mentality. These walls are perhaps seen most often between software developers and testers. One example of this is when a development group seeks to impose its way through intimidation to a user group, especially user acceptance testers. The user acceptance testers may feel that a particular issue needs to be resolved before going live with a system or release. However, the developers in an effort to meet the deadline may try to intimidate the users into living with the problem until the next release. Regardless of how valid the arguments on each side of the issue are, the fact that there are two opinions means that people will join a side. Some people will hold fast to their positions, even at the expense of the project success. The problem then becomes larger than just the resolution of the particular issue at hand. People are now pulling different directions on the project and the overall goal of delivering a quality system tends to get lost in the discussions.

Getting One's Way At The Expense Of The Project Or The Customer

This type of negative impact is seen often in testing. An example is when one person wants the application or system to go live regardless of the information from testing. I am very much a proponent of management making the implementation decision based on information from many sources, including testing. The problem is when one person forces a bad project decision through intimidation. Unfortunately, in these kinds of situations many people suffer innocently at the preferences of the intimidator.

Dividing Loyalties

People should be heading toward a common goal on a project. However, when sides start to emerge and intimidation is used as a means to "win" a disagreement, people will naturally gravitate toward the leaders they agree with. Even more significant is that people will reject or at least minimize their support for the leaders they feel are exerting pressure. Many times, these loyalties follow functional lines, such as developers, testers, users, etc. However, divided loyalties are also seen within groups, as opposed to teams. I make this distinction because divided teams are dysfunctional and typically don't last too long in a divided condition. The nature of a team is to be unified. In a divided group, such as a test organization, people may be loyal to the intimidating leader or to the people in the group who are being intimidated. One example of this is when the test leader resorts to intimidation to get people to work on Saturday. The testers may give in to the intimidation, but if they don't want to be there the leader has just lost some influence. It won't take too many Saturdays at work before the leader's influence has been spent and test group starts to explore other work options.

Cutting Off Communication

Effective communication is essential for all aspects of a project, and trust is the basis for good communication. If people can't trust that their information is "safe" with the receiver, then they aren't going to share it. When intimidation is used as a means to get one's way, trust is one of the first things that is lost. When someone starts to intimidate, it means that they prefer to use force over other more positive forms of influence to get the job done. Force is easier to use because the intimidator doesn't have to spend time understanding the other person's perspective – "Just do what I say." Communication between developers and testers is essential for the smooth flow of information and work products. When the developer to tester communication bridge goes down, so does project productivity.

Removing Focus From The Project Objectives

When intimidation comes into the project, people become so upset over being forced to do something, they spend more time and emotional energy brooding over the intimidation rather than the project or even the issue itself. People start to dwell on interpersonal problems rather than getting the job done. This can be seen in any number of ways, but break times and lunch are especially revealing of the interpersonal aspect of a project.

Creating A Culture Where Things Are Hidden (Or Things Are Revealed To Manipulate The Intimidator)

In organizations where intimidation is seen often, a culture develops where it is a virtue to hide information. This goes back to the issue of trust discussed earlier. However, intimidating cultures have trampled out trust so long ago that lack of trust has turned into fear. Hiding Information is a common occurrence between developers and testers when levels of trust erode. The lines of communication in these kinds of situations can become very complex and start to resemble a soap opera more than a project. In fearful and closed cultures, people learn the nuances of who they can trust with which level of information and how likely it is that the information will be used by someone for intimidation in the future. To make things even more complex and bizarre, information is sometimes revealed to a known intimidator for the sole purpose of seeing that person intimidate someone else. This takes the culture down to the high school level.

Creating An Over-Emphasis On One Person's Or Groups' Perspective

Intimidation can also give a false impression of a situation just by the force of the person giving the information. I encountered this once in a week-long session to identify the functional responsibilities at a company that was making their second attempt at developing a new company-wide application. There were over a dozen functional areas that needed to be addressed, such as accounting, order processing, human resources, customer service, etc. Each functional area listed the activities it performed. As the meeting progressed, it started to appear that the customer service area did more activities than any other group. However, as I took a second look at the lists, I realized that this was largely due to the fact that the head of the customer service department was a very forceful person and dominated a lot of the discussions. The other people in the group from other departments were not able to contribute their activities. So, we had to address the functions in smaller groups to get a more accurate idea of what each department actually did.

How Intimidation Can Impact Software Testing?

Most people will agree that intimidation is detrimental not only to the project but to people on the project. Software testers often seem to feel the impact of intimidation in a variety of ways.

Convincing Testers that a Defect is Really not a Defect

When I first started performing independent testing many years ago, this was one of the first occurrences that became a shocking reality to me. I would report problems, but people wouldn't believe me. It's not that they were accusing me of lying, it was that they believed that I must be "doing something wrong." I learned that I had to carefully trace my steps and be able to show someone else what I saw. The intimidation really came into play when someone in the development group would try to convince me that, "yes, we see that the situation really happened, but it's really not a problem." If I went along with the "no problem" assessment, then it was my reputation on the line if the problem occurred in the real world use of the software. If I stood my ground and insisted that, "no, it really is a problem" that's when the intimidation started, especially if the deadline was near.

Overriding or Minimizing the Results of Software Testing

Sometimes a manager, project leader or customer wants so badly to see a project implemented at a certain time for a reason that they choose to take a risk and implement the project even in the face of negative test results. The implementation decision should be a team decision based on information from testing, customer support, technical support, operations and any other area that will be involved after the project is implemented. The intimidation factor comes in when one person or a group of people try to control the implementation decision, although others see the downside risks. In projects where the implementation decision is not team-based and is between the project manager and the QA manager, intimidation can be seen in very quiet, yet powerful ways. When the project fails, people may wonder why the implementation decision was made, but few will understand the intimidation that occurred behind closed doors.

Convincing the Testers that the Deadline is the Most Important Project Milestone

Many times on projects the deadline looms as such an immovable object that people can't see anything else. People are afraid to even suggest the possibility of missing the deadline to deliver a higher quality system. The reason the deadline looms so large and immovable is because someone has communicated it that way. Certainly, there is a need to manage to timelines and milestones. Otherwise, a project would go off track quickly. However, to make the deadline the only target is to miss other important goals, such as quality and scope. Intimidation is seen when customers, uses, or project managers keep exerting the force of the deadline to continue the project death march – "We've go to make the deadline or else…" becomes the project slogan. Some managers set aggressive deadlines for the sole purpose of seeing how much work the people can do in a period of time. Another motivation for aggressive deadlines is to motivate a team that may have started to perform slowly.

Blaming the Testers for Poor Software Quality

Some people in organizations have a misunderstanding about testing that results in blaming testers for excessive levels of defects they might find during testing. These people fail to understand that the testers are inspectors and are not responsible for the initial quality of the product. If testing is performed as an end-project activity, the defects found will be higher than if a life-cycle approach to testing is performed. Performing most tests at the end of a project also results in many fixes being applied just before implementation. The nature of most systems is that one change can have a ripple effect throughout the entire system, possibly impacting other systems as well. The more defects found at the end of the project, the more chaotic things become and the most natural solution that occurs to many management groups is to simply "stop reporting so many problems!" Of course, this is not a solution at all, just delaying the discovery of defects until the customer starts to use the system.

There may be people on the project that see the folly in reducing the intensity level of testing and seek to press on to continue finding problems. It is at this point that the pressure of intimidation can be seen in an effort to get those people who want to persist in testing to "lighten up." If the testers relax the effort, they will also probably be blamed later for not finding defects that the users encounter. If the testers persist in testing and reporting problems, they risk being ignored, replaced, or worked around for the goal of delivering something by the deadline.

How to Deal with the Intimidation Factor

Although we have seen in the above discussion how intimidation can negatively impact a project, especially in software testing, the good news is that there are things you can do to deal with intimidation.

1. **Understand that intimidation is a natural occurrence on a project and prepare yourself mentally for it.** People tend to use intimidation when they run out of other ideas. Just understanding that intimidation will likely occur is a step forward. Having a mental strategy of how you will respond to intimidation is a helpful way to diffuse it. Remember the illustration at the beginning of this article about how a martial artist may yell loudly at the start of a match to intimidate their opponent. What if the opponent expects the yell and responds in their mind "OK, I was expecting that. I think I'll try a kick to the head." By anticipating the intimidation, you have taken away the surprise and therefore, the power of the intimidation.

2. **Understand that you may not be able to change the intimidator, but you can control how you react to them.** The intimidator deals mainly in fear of future events, for example the "or else" threats. You can take away the intimidator's leverage if you don't care want happens. Remember the bully from grade school that threatened to beat you up if you didn't do something? You had a choice – either give in to the bully and live to be intimidated again, or to stand firm and see what they will actually do. Standing firm has a price. You have to

be prepared to live with the consequences of not being intimidated – such as losing a job or the loss of a working relationship with someone you thought was a friend. The thing to remember is that you have control over who and what intimidates you, and that is a freeing realization! Another positive outcome is that standing up to the intimidator often makes them think twice before intimidating you in the future.

3. **Understand that intimidation is an emotional response, based on feelings that you allow the intimidator to manipulate.** Once you can feel empowered to do your job regardless of what the intimidator thinks or does, then you can be free of their negative influence. Sometimes you can be intimidated by what you think someone else is thinking. This kind of intimidation may be unintentional on the part of the other person and can be a totally irrational response on your part. As humans, we are emotional, some more than others. Regardless of how much we let our emotions show, we still live with feelings whether good or bad. The problem is not with our feelings it's how much power we give to our feelings.

4. **Educate the organization on effective interpersonal skills, including communication on projects.** Conduct workshops on intimidation and make sure known intimidators attend. These workshops are very valuable to both the intimidator and the people who are often vulnerable to intimidations. You will need a skilled facilitator and instructor to lead small groups of 12 or less through interactive role-playing exercises.

5. **Educate the organization about what testing can and cannot do.** Also, educate people about the advantages of life-cycle testing vs. testing projects at the very end.

6. **Reduce the emotion-based problems by using process-driven methods.** The decisions and actions that intimidators often try to impose are instead dictated by process criteria, not emotional-based decisions. Processes can do a lot to take the politics out of routine project activities, such as test reporting and giving the bad news. I like the example of what happened to a friend of mine when someone tried to intimidate their way into getting an "emergency" change placed into production. My friend, as the configuration manager, asked the requestor, "Is the problem your change is addressing: 1) a safety problem to our employees or customers, 2) a legal or regulatory problem, 3) a cause for loss of business?" The requestor could not justify any of the criteria had been met, so to adhere to the release process, the emergency change wasn't granted. The decision was made by the process. Had the process been violated, a precedence would have been set as to weaken the entire release process.

Conclusion

Intimidation is a part of just about every project and it seems that testers are in a position to deal with intimidation more than any other group in a project. The good news is that there are ways to deal with intimidation and keep the project on track, even in the area of software testing. The key is to understand that each person has the ability to control how much power they give to the intimidator.

Amplifying Your Effectiveness

Edited By Gerald M. Weinberg, James Bach & Naomi Karten

Format: Paperback, 160pp.
ISBN: 0932633471
Publisher: Dorset House Publishing
Pub. Date: January 2000

★★★★

buy it now

Overview

The main audience of this book would be I.T. Managers, Project Managers, and Software Test Team Leaders.

This book is a collection of essays from software professionals from various industries. The editors skillfully chose essays from these professionals to get the maximum information from their vast experience.

The book is compiled in to four sections:

- Part One: Empowering the Individual
- Part Two: Improving Interpersonal Interactions
- Part Three: Mastering Projects
- Part Four: Changing the Organization

Part one explains that the individual must be empowered. There is nothing worse than laying the responsibility on someone's shoulders and not giving them the power to make it happen.

Part two explains the importance of interpersonal relationships which I like to call “working and playing well with other”. Now that you have the power – how do you use it?

Part three moves right in to mastering the project and not allowing the project to master you. This not only tells you how to control the project but also how to recognize the signs of a project going out of control before it is too late.

Part four gives direction on introducing your newfound knowledge into the organization. The essays point out resistance to change and how to overcome this through change management.

In conclusion the software world, development and testing, can be chaotic. This group of essays will help the reader learn from the past experience of other IT professionals to avoid these situations in their project.

I recommend this book.

Scoring

Readability - 5
Breadth of coverage – 3
Depth of discussion - 3
Accuracy - 5
Credibility - 5
Organization - 5
Overall Score – 4

Major Topics

Part One: Empowering the Individual

The Role of Testing
- James Bach

A Brief History of the Accessibility of Computers by Blind People
- Kevin Fjelsted

Solving Other People's Problems
- Don Gray

The Perils of Parallel Projects
- Johanna Rothman

Do I Want to Take This Crunch Project?
- Sharon Marsh Roberts and Ken Roberts

Part Two: Improving Interpersonal Interactions

Life as a Software Architect
- Bob King

Step One in Building Strong Business Relationships
- Naomi Karten

Congruent Interviewing by Audition
- Gerald M. Weinberg

Maneuvers to Disable a Team
- Becky Winant

How to Deal with Irate Customers
- Naomi Karten

Part Three: Mastering Projects

Ten Project Haiku
- Rick Brenner

It's Just the First Slip
- Johanna Rothman

Quality Begins at Home
- Brian Pioreck

Managing Your ERP: How to Avoid Common Pitfalls of Implementation
- Marie Benesh

Recognizing Runaway Projects
- Eileen Strider

Part Four: Changing the Organization

The Satir Change Model
- Steven M. Smith

Modeling Organizational Change
- Esther Derby

How to Create a Process for Developing Useful Scientific Software
- Patricia Medvick

Good Practice Hunting
- James Bach

Reviewer
Carl Chandler

Automated Test Tools – Should We Or Shouldn't We

by Carl Chandler

If you are thinking about test tools then here are some things for you to consider:

A test tool is any vehicle, manual or automated which facilitates testing. Even something as simple as a manual test script can be seen as a test tool. However, when most people talk about test tools, they are most likely talking about automated capture/playback tools, which are the focus of this article.

There are risks on both sides of the manual vs. automated testing argument.

Risk of Not Automating

- Lack of precision
- Lack of test coverage
- Tester burn-out
- Inability to perform certain types of testing (load and regression)
- Labor intensive with lots of paperwork

Risk of Automating

- Improper tool application
- Learning curves (complexity of use)
- Over reliance on tool

Test tools do not exist in a vacuum - they require a well-established testing process in place for it to be effective with people trained to use it and willing to use it.

Major Issues

- You must be able to test manually before automating
- Capture playback does not tell what to test, you must still write test cases
- Introduction of tool to an existing project adds an element of chaos and risk
- People must be trained and motivated to use the tools.

The thing to remember is that tools can be a great help to your testing process but they cannot do everything. The analogy I like to use is that if I had Tiger Woods' clubs, probably the best in the world, and his clubs are his tools, I would still be a hacker. On the other hand, if Tiger woods used my clubs, which are not the worst in the world but are not on par with his, he would still be phenomenal. So the moral of the story is, a tool is only as good as the person using it. The same is true with test tools.

Next month, Randy's lead article will be *Surviving the Top Ten Challenges of Test Automation*, in which he will describe the ten most common obstacles to making test automation a reality in most organizations. If this is an area of interest or need for you, you will not want to miss next month's newsletter!

Links…

A comparison of the most common and popular automated tool sets by Ray Robinson. A great resource if you are involved in a tool search.

http://www.qadownloads.com/ftp/misc/papers/toolcomparison.doc

Macro Scheduler 6.2 - cheapware
http://www.mjtnet.com/index.mv?main.html

XSLT Test Tool - Free
http://www.netcrucible.com/xslt/xslt-tool.htm

ExamDiff Comparison Tool - Free
http://www.prestosoft.com/examdiff/examdiff.htm

Rice Consulting Services, Inc.
P.O. Box 891284
Oklahoma City, OK 73189
405-793-7449
405-793-7454 FAX

Coming to Oklahoma City!
January 15-17, 2002

Process Documentation – Part II

by Suzanne Chandler

Introduction

This month we are continuing our discussion on process documentation. Last month we looked at:

- All companies have processes even if they are not documented.
- When documenting processes one should keep an eye toward the future of the business market and the computer industry.
- The importance of identifying the root cause of the problems when attempting to document processes.
- Eliminating as much of the process failure, and scrap and rework as possible.
- The economic impact of software testing.
- The problem with the big bang approach to testing.

The items I would like to cover this month include:

- Who should be responsible for driving the process definition efforts?
- How can I get people to cooperate to write these processes?
- Where can I find a template or example of a good process?
- How much detail is needed in a process on a first cut?
- How many resources should be dedicated to developing processes?
- How do we maintain processes that change often?
- Who should be responsible for driving the process maintenance efforts?

According to software metric studies by many researchers over the past 30 years, poorly defined requirements account for more than half of all software defects. Since process definition/documentation and requirements gathering/documenting go hand-in-hand, the question is - how do we ensure we are properly gathering and documenting our processes and requirements? This must be done through testing the processes and requirements to ensure they are properly documented and maintained. Testing should begin at the very onset of process and requirements gathering.

Who should be responsible for driving the process definition efforts?

Senior management, project management and Quality Assurance (QA) are responsible for formalizing and documenting processes. One productive method for incorporating users and customers from the beginning of the process and utilizing them in an on-going process improvement effort is for the QA team to build a team of process owners throughout the organization who have an understanding of the current process, are organized, and are flexible enough to look to the future for process improvement.

The responsibilities of a process owner would include documenting and maintaining processes for a specified area and looking for ways to improve processes. Their area of responsibility should include the area they are most familiar with and work with on a daily basis. The team should meet at least quarterly to look for ways to improve their processes and to test change impact. With each process the customer should be included to provide feedback and suggestions.

Employees should also take the responsibility of identifying ways to improve processes and contacting the process owners. Including employees by soliciting their suggestions encourages a team spirit, which is the first step to success. The QA team is responsible for taking the lead role in facilitating all meetings as well as the overall process definition documentation, maintenance and consistency. They are also responsible for keeping upper management abreast of their progress and obtaining upper management sign-off for recommended changes.

How can I get people to cooperate to write these processes?

The most difficult factor in any activity that involves people *is* the people. Lots of people, with many agendas, opinions, and emotions can be intimidating and hard to deal with. Keep in mind that an organization is merely a collection of processes. Each process is made up of people and a product. And as important as the people and the product are, they are not as important as the process. Why? Because the process is what must be constantly improved in order to succeed as a business. Here's another way to look at it - However, you can never lose sight of the importance of the people or the product since the people make it happen and the product is what attracts the customer.

If you don't have a good process that is constantly improving, you won't have the people or the product very long.

So what is the solution? First, stay focused on the process. Second, focus on the improvement of the process. Third, explain to every person you deal with that the goal is to document and improve the process to ensure job security through a better system and that the goal is not to eliminate jobs.

Probably the most difficult hurdle I face in documenting processes is the fear factor of each employee I speak with. This is especially true of the front-line employees who may be afraid that if I know what they know, I will either eliminate the work and thus eliminate their job, or I might change their working method and as we all know, no one likes change!

As much as it is in your power, try to put employees at ease. However, it is management's ultimate responsibility to demonstrate that the documentation is a positive thing. That is, to enforce the cooperation of each individual, and to teach their employees that we are all working for one common goal - a better and more productive business. Work smarter – not harder!

Many times, the purpose of documenting processes (why else would we do it – unless we absolutely had to) is to begin gathering requirements for building an automated system. There are, believe it or not, many people who are opposed to automation. Their fear is the same as those stated above – loss of their job or a change in the way they currently work. I'm still surprised at the number of people who do not want a computer and are afraid of them. So, we are really dealing with two issues with the same symptom – non-cooperation. Besides management input you can also overcome these objectives by holding training sessions for the employees to help them understand the process they are going through and the goals. Attempt to include those with real fears of the computer in user acceptance testing so that they feel they have a leg up when the system is released instead of feeling inadequate and behind the times.

Randy's article this month on ***Dealing with the Intimidation Factor in Software Testing*** has some excellent recommendations for dealing with intimidation and understanding its impact on the project.

1 in 5 workers complain they don't know what is expected of them, they don't have the materials to do their jobs and they can't get the attention of their bosses. Based on these numbers, salaries and lost productivity cost businesses over $300 billion a year. - The Gallup Organization

Where can I find a template or example of a good process?

Documenting processes is very important. They must be clear and concise. If more than half of projects fail because of poorly documented processes and requirements then one must recognize that it is crucial to take the required time, utilize the best tools and templates, and test the process and documentation from beginning to end in order to start the project on the right foot.

Check out these links for some helpful tools:

http://www.construx.com/doc.htm - Templates for technical documents, and planning and management documents.

http://www.rcc.ryerson.ca/rta/flowchart/software.html - A brief listing of some of the more popular flowcharting programs.

http://www.records.nsw.gov.au/publicsector/DIRKS/exposure_draft/Interviews.htm - Guide to interviews including templates.

http://www.riceconsulting.com/requirements_course.htm - An excellent basic course in understanding the process of gathering, defining, testing and managing user requirements.

http://www.asset.com/stars/loral/process/guide/main.htm - A practical guide on how to document processes.

http://www.riceconsulting.com/inspections.htm - A practical team-based seminar to show how to perform effective walkthroughs and reviews of project deliverables.

http://www.riceconsulting.com/sitwb.pdf - A sample systems integration test workbench.

How much detail is needed in a process on a first cut?

Set standards to follow throughout the process documentation. Standards will add consistency to the process and define deliverables so that everyone is aware of what is to be accomplished, to what extent, and when it is to be completed.

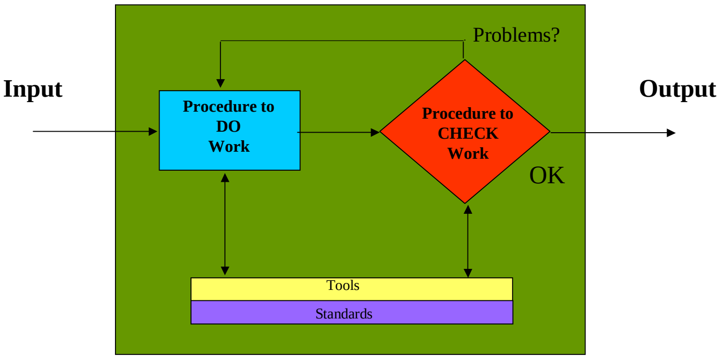
Identify any existing processes to be used to build from and be sure these are tested at each meeting level. In addition you need to identify where you are, where you want to be, and how you plan to get there.

All the best intentions will not get this project off the ground, so what is required to get us there?

- Strong management directive
- A major investment of time and money
- Cultural buy-in

For your first meeting, start with a high overview to include the structure of the department(s) to be documented and their general responsibilities. Why begin at the beginning? Because you can only eat an elephant one bite at a time. Otherwise, you will get overwhelmed by the project. Use your high-level report as a tool when meeting with an inspection team made up of management level stakeholders who will do a review of the process documents. At this meeting the stakeholders should:

- Explain the role of Quality Assurance
  - Manages quality effort
  - Defines processes
  - Monitors process effectiveness
  - Consults on quality issues
  - Encompasses testing, standards, and measurements
- Explain the role of Quality Control
  - Inspection
  - Producer role
  - Seeks to find defects for the purpose of correction
- Review the essential elements of a process
  - Input
  - Output
  - Procedures
  - Quality Control
  - Tools
  - Standards
  - Measurements



- Explain the workbench approach
  - Designed by the Quality Assurance Institute to describe processes
  - Contains all the essential elements
  - Can be used to describe many types of processes
  - The term “workbench” is more accepted than “process”
- Teach how to build a process
  - Step 1 – Assign responsibility
  - Step 2 - Define process objectives
  - Step 3 - Identify workbench components
  - Step 4 – Document the workbench
  - Step 5 – Review the workbench
  - Step 6 – Pilot the workbench
  - Step 7 – Make adjustments
  - Step 8 – Publish the workbench
- Explain the critical success factors

- Realize that all processes are imperfect initially
- Continuous improvement
- Cultural acceptance

Meeting minutes should be documented and distributed to all participants for sign-off. A meeting with the employees assigned for the next level of detail should be set up and participating employees notified. If an assigned employee is unable to attend a representative from the same area with equal knowledge should attend in their place. It is important that all areas are represented at the meeting. Follow the same steps as the first meeting using the details of that first meeting as a launching pad into the next level of detail. In this process you will confirm and correct any ambiguous or erroneous processes from the previous level. This process should be continued until all details are documented. Remember to include sign-off copies at each level to management for buy-in. Because time is always of the essence with any project, it is imperative that schedules are set and adhered to by all parties.

Rice Consulting Services teaches the details of this process in an easy to understand manner with practical applications in the following courses:

- Basic Training in Software Testing
- Building an Effective QA Process for Ongoing Validation
- Structured User Acceptance Training
- Gathering, Defining, and Testing User Requirements

How many resources should be dedicated to developing processes?

Obviously this will depend on the project size and complexity. But the team should consist of Quality Assurance personnel, Quality Control personnel, and stakeholders.

How do we maintain processes that change often?

Maintaining processes includes accounting for, controlling, and reporting processes from beginning to end. And since processes never stay the same, but are constantly changing and hopefully improving, you will be continually maintaining these processes. Your team of QA and process owners should meet quarterly as discussed above to ensure process change recommendations are discussed, analyzed, and approved to ensure the change is in the best interest of the company.

Once a process change is approved all documentation distributed throughout the company should be standardized.

Who should be responsible for driving the process maintenance efforts?

Quality Assurance (QA) is responsible for maintaining the processes with the help of the team of process owners. Management is responsible for creating a culture of quality improvement within the company and supporting the improvement team by empowering them to implement changes.

Conclusion

Discovering and documenting processes can seem overwhelming at first glance. By taking one step at a time and using good organizational and people skills you can get your processes documented and have a great start for maintaining and improving your processes. When done properly and adhered to this should lead to a more productive and efficient business.

Rice Consulting Services' Consulting Offerings:

Testing Assessments

Rice Consulting Services' testing assessment is a quick and effective way for an organization to determine where they are in terms of software testing maturity. The assessment looks at three areas that are critical to testing:

• **Test organization** - Who performs testing, what levels of experience are present, and when testing is performed in the development/maintenance life cycle.

• **Test process maturity** - How well-defined, well-deployed, and repeatable the test process is, and whether it incorporates good testing management, practices, tools, and techniques.

• **Readiness** - An assessment of the organization's readiness to improve the testing process. This involves an assessment of the staff's testing awareness, testing skills, and motivation to change current practices.
The deliverable is a report detailing the assessment's findings, a recommended quality improvement strategy, and a plan for addressing the improvement needs identified. If the assessment uncovers the need for in-house skills training and consulting, we will include

proposed training and consulting plans in the report. The report is typically about 15 pages in length.

In-House Software Testing Certification Programs

There is a lot of interest in certification programs for software development and software quality. There is also value to both the individuals being certified as well as the organizations that employ them. As you examine the various certification programs that are available, you need to ask:

- How recognized is the certification?
- What is the basis of the certification (i.e., what does it cover?)?
- How is the certifying organization accountable and responsive to its members?
- How closely do the certification criteria reflect the items important to your career and organization?
- What is the required investment to get and maintain the certification?
- What is the future for the certification program?
- What is the initial cost to your company to get certified?
- What is the annual cost to your company to retain certification?

After examining the above questions, some organizations have determined that the best certification program may be their own. One of the greatest advantages of an in-house certification program is that you can control the criteria, future and investment of the certification. As for objectivity, there are options that allow you to administer the in-house program while an independent organization verifies certification criteria.

Rice Consulting Services, Inc. has been working with several organizations recently to develop this kind of program. **We have extended certification training programs of 10, 15 and 20 days in length. These programs are tailored to your people, business, technologies and tools.** Participants range from experienced testers and QA personnel to people just entering the field.

This type of program makes the training effort more than a "one shot" event. People are tested at the end of each major topic area and are also evaluated by direct observation during exercises. The certification is normally determined by a combination of demonstrated proficiency during training as well as actual work experience. The certification criteria are defined by you, but we can help you with templates and examples.
Each in-house certification program is different. For details about how we can help you design and conduct an in-house certification program that is right for you and your organization, just call Carl Chandler at 405-414-6759 or email us at carlchandler@riceconsulting.com.

Rice Consulting Services' Course Offerings:

For those who hold professional certifications, such as the Certified Software Test Engineer (CSTE) and Certified Quality Analyst (CQA), each hour of instruction counts for one CPE credit. 40 CPE credits are required each year to keep these certifications current.

If you would like to learn more about the information covered in Carl's article we at Rice Consulting Services, Inc. offer an excellent course that will enhance your company's software quality process.

Build Your Own Course

– 2 – 20 days

Because all of our courses are designed to be modular, we can easily customize a course for you for presentation at your facility! A typical course day is 6 to 7 hours of instruction.

We provide a listing of all of our course modules at http://www.riceconsulting.com/build_your_own_course.htm. Simply select the modules you would like to have presented to your team. We provide a brief description of each module but if you would like to see more details, just click on the Module ID link. Upon submitting your course design, we will get a copy of your selections and will contact you by e-mail and phone.

Please note that our modules are not in alphabetical order. They are in order of popularity and typical presentation order. Specialized topics, such as Web-Based Testing, Client/Server Testing and E-Commerce Testing are found at the end of the list.

Structured User Acceptance Testing

– 3 day

This is one of the few courses available that teaches a non-technical and easily learned process for testing computer systems from a business process perspective.

This is a practical hands-on seminar to convey effective methods to plan and conduct user acceptance testing. This is one of the few courses available that teaches a non-technical and easily learned process for testing computer systems from a business process perspective. This course deals with testing issues from both the process and human perspectives. You will learn the terminology, the unique issues, and the process for performing user acceptance testing. As a result of attending this seminar, you should have a good working knowledge of what it takes to plan and conduct a very effective user acceptance test in your own organization.

Structured User Acceptance Testing will help you become more comfortable and confident in designing and performing a test that models how an organization will use a particular application to conduct business. You will emerge from this two-day session knowing how to develop test scenarios, test scripts and test cases. You will also have a working knowledge of how to coordinate all of the aspects of a user acceptance test into a smoothly flowing test.

Whether you are planning to test a vendor-developed or in-house developed applications, the process and techniques covered in this course can enable you to identify the most effective tests and maintain a high level of test coverage.

User acceptance testing does not need to be overwhelming and intimidating. Learn the issues and processes for effectively testing business processes by attending this hands-on course.

A Three-day course in User-Oriented Practices for Delivering Quality Software

- 3 days

Now, more than ever before, more responsibility is being placed on software users to define and validate the systems they acquire. This three-day course presents two important sides of user involvement in software projects: gathering and documenting user requirements, and testing from the user perspective.

This is a practical interactive seminar which uses team exercises to reinforce the process taught in the class. You will learn the terminology, process, and challenges of requirements management in the real world. As a result of attending this seminar, you should have a good working knowledge of user requirements and what it takes to gather, design, test and manage a complete set of user requirements for a project. In addition, you will learn how to effectively plan, perform and evaluate a user acceptance test.

This program will help you become more comfortable and confident in performing the requirements management process in just about any role on the project, including business analyst, user, system designer, developer, project manager, QA analyst or tester.

You will emerge from this three-day session knowing how to define the right problem, talk to the right people, document the right needs, build the right system, and test the system using a defined baseline of requirements and business processes as the target. You will also leave with a knowledge of how tools can help you perform requirements management.

For more information on this course or one of our many other offerings please contact Carl Chandler at (405) 414-6759

Rice Consulting Services, Inc.
P.O. Box 891284
Oklahoma City, OK 73189
405-793-7449
405-793-7454 FAX

Coming to Oklahoma City!
January 15-17, 2002

Frequently Asked Questions

by Randy Rice, CQA, CSTE

Q: Thank you for your suggestion regarding UAT! I though you might be interested to know how everything worked out.

As you suggested, I created a script for them to follow (3 pages) – basically "perform a task" and told them which screens to use (because their workflow was changing) - but not exactly how to go about doing it. I also included some fill-in-the-blank questions (for example, "How many requests did we get from customer ABC?") - again telling them which screens to use to find the information.

We invited 5 "real" users and 3 of their managers and kept the session to 1 hour. We held it in our training room - everyone had their own PC to use. We had set up some test data in the test environment before the session, but didn't limit them to just those records. Our tech support and testing staff were there. The development team leader and our technical trainer also attended.

Since this group had never done a session like this before, I had to explain that this was not training - they were not expected to walk out of there knowing how to use the system. I told them that they should start with the script, but not to feel limited to it - it was OK to go off on a tangent if they wanted.

The users loved it. They liked having a script to follow, because in the past they were just told to "play" with the system in the test environment - they had no idea what to do. With the script, they had a place to start. When they ran into trouble, we were there to help. They wrote comments all over their scripts.

I liked it because it gave us good feedback on what might be confusing and what needed to be included in the training sessions. The users found some bugs that our test scripts missed. We also got some really good enhancement requests - many that we were able to include in the release. The user managers liked it because we could involve twice as many people in reviewing the software before release, and they were able to talk through the new workflow during the session. The development team leader was able to address any problems that came up, and offer suggestions in response to user requests.

So our first UAT was a success! In fact, the users insisted on scripts to use after their "official" training sessions so they could practice before the new software went live!
Thanks again for some great advice!

A: Thanks for letting me know how things worked out. I'm really glad to hear of your success!
(Editor's note: This was a response based on a previous question, which inquired about how to perform user acceptance testing in an organization that probably wasn't ready for rigorous methods of UAT.)

Q: I am an associate quality assurance engineer and would like to automate usability test cases. Would you kindly give me input regarding the points to be kept in mind for automating usability test cases.

A: Basically, usability testing is not a test that should be automated. The point of usability testing is to determine the ease of use for a typical user. This requires manual testing, often with little or no coaching.

Q: Hello! I wanted to let you know that your site is very helpful! There is a particular skill that I'm searching for, QA Run. It's a specific testing tool (apparently very hard to find!).

A: Yes, QA Run is a capture/playback tool sold by Compuware. Compuware is now more heavily promoting their newer tool, Test Partner. I like QA Run for doing cross-platform character-based testing.

Q: If code is reusable then should not testware be reusable as well? How could testware be designed to maximize reuse? I hope you can help,

A: Thanks for the question. There are three ways to achieve testware reusability that I like to use a lot. The test tool vendors also promote these techniques.

1 - Modular test cases and test scripts. Instead of writing long test scripts that perform many functions, break the scripts into separate functions and combine them into many different combinations. Using this approach, which is very object oriented, you can treat the test script or test case and the function and send all kinds of data through it. You can create many tests from a small number of cases just by applying different data sets.

2 - Separate test data from test cases and test

scripts. By keeping the function separate from the test data, you reduce testware maintenance and increase flexibility. The data drives the test.

3. - Hierarchy of test cases. To understand which cases you want to test, start at the top and keep breaking down. Example - The top level entity would be a customer. The next level would be types of customer:

Regular, Preferred and Former. Within each of these types would be locations: Domestic and Foreign. This process continues until you reach a very low level of detail, then you have the basis to design test data.

These three techniques work together very well. I hope they help you.

Q: I've read your article on Acceptance Testing. I thought, maybe you can help me: I am looking for materials on Acceptance Testing of Billing Systems. If you have any articles or know where to find them, please let me know. I will appreciate any help.

A: When it comes to billing systems, timing is a key thing to test. You want to be able to simulate the passage of time as the test proceeds. I wrote an article about this for the testing of date-sensitive processes for Y2K, but the technique is just as applicable today as it was then.

The foundation of the test cycle approach is that you define simulated periods of business activity that makes sense in your applications. For example, you may want to conduct tests that simulate 14,15, 16, 29, 30, 31 days out from a certain date. These are common threshold dates for billing systems. You might have others or different ones.

As for other articles, they are few and far between. We have been doing research and projects in UAT for over 10 years now and have a complete course on the topic. However, I don't know of any books on the topic.

I hope this helps!

Q: What type of information would be in a test strategy and not in a test plan?

A: Test Strategy - A brief high level document that describes the overall approach, objectives and direction of the test. The test strategy can be written very early in a project and does not require the knowledge of specifics, such as people names, modules names, test tool brands, etc. The test strategy is a generalized document that communicates to the rest of the organization how testing will be performed. In contrast, a test plan builds on the information contained in the test strategy and describes the logistics of a test, such as who will be on the test team (by name), when the test will occur (by dates), where the test will occur, the order of testing, test objectives, pass/fail criteria, test reporting and any other project considerations for the test.

Notable Quotes…

"You can't accomplish anything unless you have some fun."
- **Charles Knight**

"The boss drives his men; the leader coaches them. The boss depends upon authority; the leader upon good will. The boss inspires fear; the leader inspires enthusiasm. The boss says 'I'; the leader 'we.' The boss fixes the blame for the breakdown; the leader fixes the breakdown. The boss says 'go'; the leader says 'let's go!'"
- **H. Gordon Selgridge**

America was targeted for attack because we're the brightest beacon for freedom and opportunity in the world. And no one will keep that light from shining.
-George W. Bush, Jr.

“Let us not wallow in the valley of despair. So in though we face the difficulties of today and tomorrow, I still have a dream. It is a dream deeply rooted in the American dream.”
- Dr. Martin Luther King, Jr.

“’Duty,’ ‘honor,’ ‘country’ -- those three hallowed words reverently dictate what you want to be, what you can be, what you will be. They are your rallying point to build courage when courage seems to fail, to regain faith when there seems to be little cause for faith, to create hope when hope becomes forlorn.”
-**General Douglas MacArthur**

“The wicked are overthrown, and are not: but the house of the righteous shall stand.”
- The Bible - Proverbs 12:7

The Reason for the Season

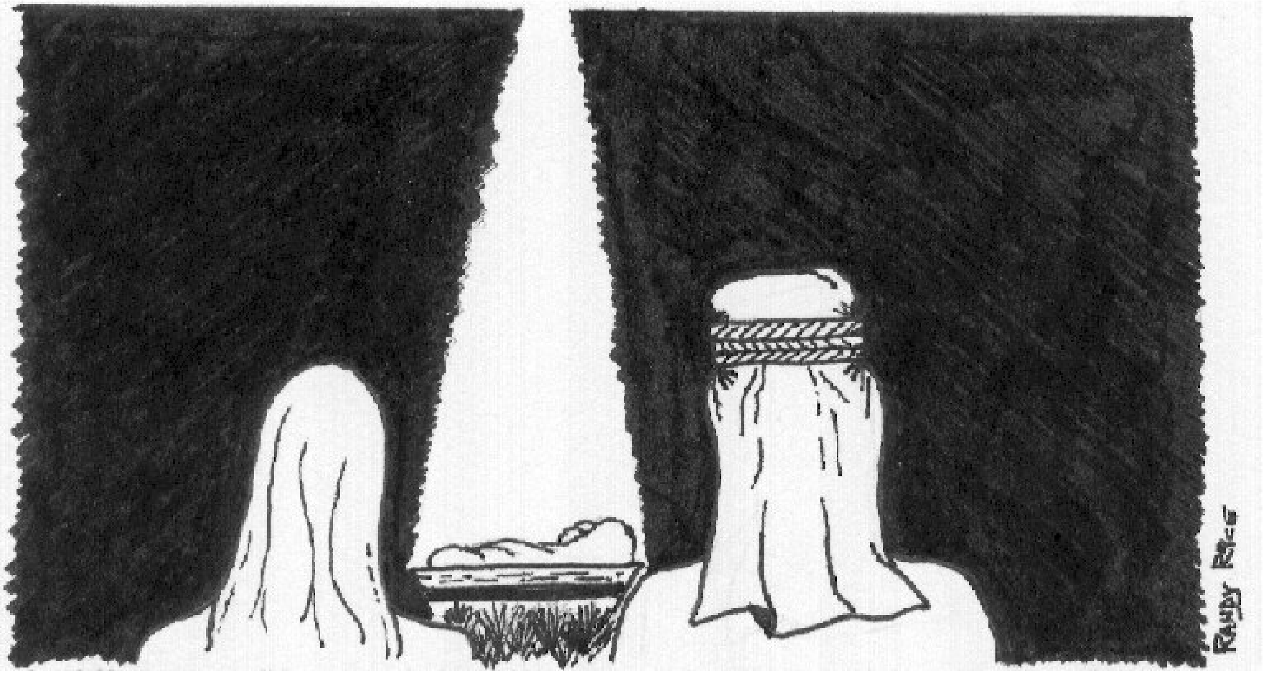
During this time of year we are busy with Christmas parties, buying gifts, hooking up our lights, visiting with family and friends. But what is the season really about?

Nearly 2000 years ago a baby was born in a manger. The prophecy of this baby's birth was before the beginning of time. This baby was not just an ordinary baby. No, He was God Himself, in the flesh, who came down from heaven to take away the sins of the world.

He was born a man to be a kinsman redeemer. For each of us owed the sin debt that we could not pay. The penalty for that debt is death. Yet our God, who created the universe with His words, chose before the beginning of time to give His life for us by being born a babe and growing up a man. He died in our place so that we might live eternally with Him.

So let us remember this Christmas the words of the angels that appeared before the shepherds, "Behold, I bring you good tidings of great joy, which shall be to all people. For unto you is born this day in the city of David a Savior, which is Christ the Lord."

Suzanne Chandler



In Him was life, and that life was the light of men. The light shines in the darkness, but the darkness has not understood it. John 1:4 – 5.

May the Light, Jesus Christ, shine on you this Christmas and bless you in the coming year!

Randy and Janet Rice
Carl and Suzanne Chandler

January 2000 Issue:

- **Surviving the Top Ten Challenges of Test Automation**
  by Randy Rice, CQA, CSTE

- **Skills of a Test Team Leader**
  by Carl Chandler

- **Testing on a Budget**
  by Suzanne Chandler

Coming to Oklahoma City!

January 15-17, 2001

To present to you a Three-day course in Web Testing Technologies
http://www.riceconsulting.com/okc_q1_2002.htm

Coming to Atlanta!

February 11-13, 2001

To present to you a Three-day course in User-Oriented Practices for Delivering Quality Software http://www.riceconsulting.com/atlanta2002.htm

Rice Consulting Services, Inc., *a world recognized leader in Quality and Testing Training.*

This certificate worth _____ CPE credits* towards Certified Software Test Engineer Continuing Professional Education through the Quality Assurance Institute.

*Category E - Self-Study Courses Activities designed to improve your proficiency in CSTE skill areas as defined in the *Common Body of Knowledge* may qualify for CPE credit up to a **maximum of 20 credits per yea**r. Qualifying activities include: Professional memberships that offer self-study education regarding quality assurance within information technology. It's not the membership that earns the credit, but the study materials provided by the membership.

To redeem complete the following information and submit to the Quality Assurance Institute at the time of reporting CPE credits.

Name: ____

CSTE/CQA Certification Number: ________________________
(circle one)

Email Address: ____

Credits available to members of The Software Quality Advisor only. To become a member of The Software Quality Advisor sign-up at http://www.riceconsulting.com/SQAdvisornew.htm

Rice Consulting Services, Inc.
P.O. Box 891284
Oklahoma City, OK 73189
(405) 793-7449 / (405) 793-7454 Fax
e-mail rcs@telepath.com
http://www.riceconsulting.com