INFORMATION MANAGEMENT

SOLVING THE SOFTWARE QUALITY MANAGEMENT PROBLEM

James L. Mater

INTRODUCTION

In its capacity as an independent software testing lab, QualityLogic, Inc. has worked with the information systems groups of both small and large companies, and with software and systems companies. This work provides a unique opportunity to observe the struggles that organizations go through in attempting to solve their software product quality and quality management problems. This article presents resulting observations and thinking about the basic quality management problem, as well as a new solution for the industry.

Profit and loss (P&L) managers, referred to here as business managers, do not clearly understand or value software quality management. Software companies and projects fail to deliver quality products because they do not treat quality management as a strategic, critical aspect of the product development process that is equal to requirements, design, and code development.

The basic problem is that software quality management is not properly “owned” within organizations. Instead, because it has historically been relegated to a software quality assurance function, it is considered technical, thus, few business managers would ever consider directly “owning” it.

PAYOFF IDEA

Software company business managers have a new option for solving the software quality problem. Software quality management can be outsourced to a competent, independent organization whose primary focus is software quality. This model provides an effective and efficient solution to the industry problem of obtaining higher quality in its products, while managing costs and creating an appropriate cultural environment for those engineers who choose software QA as a career path.
In traditional industrial businesses, product quality management, quality assurance, and quality control are treated as major corporate functions, reporting to the business manager. However, few software organizations have yet adopted this approach. Indeed, the software business is such a recent “discipline” that the issue of product quality management remains a mystery, especially to business managers without software training or experience.

The software quality issue — which ensures the quality of software products delivered to customers — is not a technical one. It is achieved through a combination of good customer understanding (developed into requirements) and good product development processes.

There are many excellent software development processes and techniques that are both proven and available. The software industry knows how to build high-quality, reliable products that meet the feature, cost, and schedule needs of customers; and that are easily maintained and upgraded. It is a myth that the industry needs better processes or tools to solve the quality problem, which skirts the real issue of business manager responsibility.

Unfortunately, the industry typically does not have that combination of discipline and organizational structure required for consistently delivering successful products; that is, a well-defined, well-executed quality management function. Responsibility for this must start at the business management level. The quality problem will remain unsolved until these managers think long and hard about the quality requirements for their products, until they have clearly communicated their conclusions, until they actively monitor product quality, and until they are willing to act on that information to enforce their policies.

SOFTWARE QUALITY POLICY

Business managers have two critical responsibilities relative to software quality. First, they must set and communicate clear policy, empowering their people to carry out that policy. Second, they must ensure that these policies are implemented. This entails monitoring quality on an ongoing basis and taking action as needed to keep the organization on track.

Business managers must give serious thought to software quality policy, answering the following questions:

- Is the organization’s policy to be first to market with the right features at the right price — and fix reliability issues later?
- Is it to have the most reliable product available in its class?
- Is it to aim at the low end of the market, which will accept poorer quality at a lower price?
• Are there critical safety or customer issues that demand perfection, in terms of 100 percent reliability? (This is the case, for example, for medical instruments, defense systems, and avionics components.)
• Is the company committed to a zero-defect policy?

The task of determining policy cannot be delegated. Only the business manager can set this policy, because all others in the organization will let the policy be influenced by their (individual) quality goals and the design/implementation of the software.

Thus, the policy should carry the weight of the business manager, reflecting serious consideration and commitment. It should have lasting value, and be unambiguous to those implementing policy.

There really is value in thinking about and articulating such a policy. Quality problems in the software industry are caused by the lack of clear direction from the business manager and the will to enforce such policy.

Business managers can only hold their teams accountable for meeting quality standards if these are stated. A decision to ship a product can only be made when there are clear criteria for making such a decision. A development team can only be disciplined for causing exorbitant support headaches when team members are told that minimizing support costs is a critical issue at the time of software design. A product manager can only set quality goals for a product when the standard corporate policy is consistent from day to day and product to product.

The right policy must be set and articulated before it can be enforced. A business manager who does not step up to this issue is negligent in leading his organization.

**MONITORING AND ENFORCING QUALITY POLICY**

Once a quality policy is put in place, the second major issue is monitoring product quality to ensure that the policy is carried out. This means that business managers must establish a good quality management function that provides their organizations with good information about the quality of the products under development, and enforces their quality policies. The policies and their enforcement have failed if the business manager only finds out that customers are dissatisfied after a product has shipped.

The proactive business manager must determine if the products in development will be delivered on time, on budget, and with the quality required to succeed in the market. Uncommon managers, who have put in place the right organization and people with the right direction, need merely ask for the information, and it will be available in some form, providing an accurate view of the quality of products in development.

Unfortunately, for most organizations and business managers, this is an unrealized dream. While there may be a test team in place to measure
product quality, it is probably buried in the development organization, wherein inexperienced staff report to an inexperienced test manager. Here, the right information seldom reaches the right people in time. Rather than an independent function, quality management is a lower-level quality control function, performed by the test team, which has minimal understanding of corporate quality policy and issues.

What the organization needs is a quality management team that:

- is independent of the development team
- is empowered with the authority of the business manager
- is working with the product on a day-to-day basis
- has the skills to thoroughly evaluate the product against explicit or implicit criteria, and can ferret out the evaluation criteria from whatever internal sources are available — or raise a flag if adequate product requirements do not exist
- can professionally provide documented information to both the development team and the business manager
- clearly understands the manager’s business problem and is helping to solve this above all else
- operates very efficiently and effectively

Unfortunately, it is difficult — if not impossible — for a business organization to put this definition in place internally.

**MANAGING THE QUALITY FUNCTION**

Product quality management is the executive function that owns the process for delivering products of the quality required by the marketplace. The function starts with good product requirements, moves to a development process that is designed to deliver predictable results based on the requirements, and ends with a quality control process (testing), which validates that the product indeed meets the defined requirements.

The development process must include explicit quality assurance steps to succeed. However, most company executives concentrate on requirements and other aspects of development, treating quality assurance activities as an afterthought.

Few organizations have a designated quality management function, although some have a software test department. Others have a quality assurance department that they refer to as “software QA,” but it is really a software test group. Invariably, and despite protests to the contrary, this “software QA” department is often the weak link in the chain. Companies manifest the symptoms of this weakness in various ways:

- **The software quality assurance function itself is typically a “hot potato,” which no senior manager wants to own.** The func-
tion is moved around from engineering to manufacturing to operations and back to engineering. It seesaws between a centralized and decentralized function every couple of years.

Two companies that QualityLogic recently interviewed, dissolved the central QA function, redeploying the engineers to the product teams and causing a great deal of disruption. Both organizations came to the conclusion that the central function was not working well after two to three years of effort to make it an effective business tool. In another case, the vice president who had been “given” QA was all too happy to hand it off to an outside company.

- **There is discontinuity in the management of the QA function itself.** It is difficult to find and keep a good manager in software testing or software QA. Instead, managers often move out of the function. If they are really good, they are often hired away for more money; if they are ineffective, they are often fired. In any case, it is rare to find stable management of the software QA or test function.

- **There is no encouragement; it is rare that highly respected developers move to software QA.** In fact, the opposite is true. Many companies are proud of the fact that they can use software QA as an entry point and training ground for development. The most attractive career path available to the QA engineer is to move to development. For example, one of QualityLogic’s major customers has a terrible time keeping good test leads. Hired right out of college, they have been screened for good development skills and are moved into development as soon as they become effective test leads. While this works well for the development organization, it continually leaves software QA with an inexperienced staff.

- **There is a constant turnover in QA staff.** The consequence is that the QA organization never matures to the same level of skill and professionalism as development teams. Companies are often proud to have a stable QA organization for one or two years. This is in sharp contrast to the stability and maturity of the development team, which has typically been the same for five years or more. Thus, the company should recognize that the QA team is not even close to adequate for the task.

- **The use of developers as testers.** A major QualityLogic customer recently needed help with a critical project. Its division management had just fired all of the QA engineers in an attempt to “fix” the quality problem. The company’s ISO 9000 model stated that developers should actually do all of the quality assurance and final acceptance testing themselves — but this group just did not have the bandwidth to do so.

  Although developers should indeed “own” the quality of their work, and should conduct such quality assurance activities as unit testing and peer reviews, they should not be the final product testers.
Developers are seldom motivated or particularly competent as final product testers. In addition, the lost-opportunity cost of pulling them off of development work is staggering, when analyzed.

- **The development engineers successfully lay the blame for quality/schedule/feature problems on software QA.** The weak link is a test or QA team that is unable to effectively advocate its own position; the team gets dumped on over and over again.

  One major company is currently debating how to fix this very problem. The organization has an excellent QA team that does system test, but it works under the vice president of engineering. Because it is part of engineering, the QA team relieves the development teams from passing all the entry criteria before products acceptance for system test. Of course, QA is then blamed when the ship date slips.

  While this situation is very typical, it is also easily solvable. The business manager must determine clear accountability for both development and QA functions, and establish a quality management function to enforce policy.

- **The QA team is unable to communicate product quality information to decision-makers — primarily the business manager.** The team might lack the experience to decide when information is critical to the business manager. Alternatively, the team’s information may be filtered through the current owner, usually a vice president of development or engineering. As a result, the information serves the VP, but not the business manager.

- **Ship dates are frequently delayed, and the delays come as surprises (at first)—to everyone except the developers and testers.** The testers did not try to make the information available to the business manager, or were unsuccessful in doing so.

- **Product design or features are routinely changed, causing schedule slips and expensive rework and retest, before release.** Management accepts major design or feature changes because the basic process discipline was not controlled from a quality perspective. No one enforced the early steps of requirements verification or design review, and the impact on quality control activities was ignored in the decision process. This happens more often when there is an inadequate quality management function in place.

These problems all result because the business manager is not adequately investing in quality management. Nor is he or she willing to insist on accountability by the development group. In many cases, the definition of “adequate” is not understood, and quality management is under-funded. Because quality in software is treated like an engineering function that no one really wants to own, it is no wonder that software QA people are also inadequately treated.
Thus, software test and QA engineering jobs are entry-level positions used as a training ground for development. Because the best people are routinely migrated to development, this perpetuates the weakness in quality organizations. An organization will have difficulty maturing when all of its members are entry level and intent on moving to development.

Furthermore, software test and QA engineers are treated as second-class citizens. They are not considered as good as developers because of a bias that suggests: “those not good enough to code, test,” or “those who can, write code; those who can’t, test.”

In addition, software test and QA engineers are poorly paid relative to development engineers, and there is little or no career path for the former. Therefore, test and QA engineers do not have nearly the same opportunity as developers to rise in grade and pay.

This inequity extends to budget decisions, which also favor development over QA. If, for example, both QA and development ask for tool sets for their functions, and the company cannot afford both, development usually wins. Finally, management is willing to let QA suffer if development slips its schedule.

All of these problems and indicators stem from the business manager’s lack of clear understanding and valuing of software quality functions. This set of problems may be seen as cultural and management challenges facing the business manager.

**SUCCESSFUL SOFTWARE QUALITY MANAGEMENT**

Solving this problem set is simple: business managers must clearly understand the quality requirements of their products, be willing to make appropriate strategic decisions about them, and then put in place a quality management function. In the past, this has meant funding an independent software quality management group that does not report to engineering, and insists on disciplined behavior during the whole process. The group is typically used as a measurement and control mechanism.

Traditionally, an executive-level vice president, director, or manager of quality probably reported directly to the business manager. This provided adequate budget, experience, and power to enforce quality disciplines, and act as a gate for product release cycles. Currently, quality is often approached by integrating the quality functions into development teams via senior quality people, and establishing a clear, appropriate process for control of quality during development. While this can improve the organization’s ability to develop high-quality products on time and within budget, it does not provide an objective, independent view of product quality to the business manager.

Alternatively, strong business managers can require that the quality function (usually just a test group) report to them directly. They can hire a vice president of quality to work directly for them, and manage the test
function. They can ensure that the development vice president also views product quality management as important and sees the need for an independent quality function.

In the end, the business manager must spend a significant amount of effort and dollars to develop a strong QA organization. Three years ago, for example, one CEO of a leading software company placed QA directly under him. Unfortunately, the QA manager was not strong enough, and a major release was shipped with significant problems. Only then did the CEO finally understand the caliber of manager required, and it took another few months to find that person. Now the company is in the rebuilding phase, and the jury is still out on the success of this approach. It is actually unusual that a business manager would make these decisions. Instead, most continue to struggle with this problem but never really solve it.

For business managers to succeed in the software business, both internal and external quality management functions require the following characteristics:

- the business manager’s clear definition and enforcement of a quality policy
- authority directly from the business manager, and independence, at least within the organization
- team stability and maturity as evidenced by pay, promotional opportunities, and team tenure comparable to development; an understanding of the business of developing successful software products; and earned respect from the whole organization
- ongoing investment in generic software testing and QA skills
- ongoing investment in tools and process improvement for the QA and test functions
- an incentive structure that reinforces both effectiveness and efficiency in the QA and testing functions

If a company spends its resources in meeting these requirements, it can and will maintain a powerful quality assurance function equal to the other elements required for product success. However, these investments are often difficult for organizations to justify, and they require sustained interest by the business manager. A viable alternative is to outsource some or all of software quality management, software quality assurance, or quality control to a third-party specialist in this area.

Outsourcing some or all aspects of the software quality management function is an emerging approach to the quality problem that has evolved naturally. This solution recognizes that the quality function must be done well, but it need not be a strategic internal competency. Quality management, quality assurance, and test comprise a discipline, complete with a generic methodology, process, and tools. Companies must deter-
mine whether it is a strategically good investment for them to outsource, or to develop and maintain this functional expertise themselves — which is an expensive proposition.

**THE EVOLUTION OF SOFTWARE QUALITY MANAGEMENT**

The business aspects of software quality are evolving, along with hardware platforms, software languages, software development tools, and the process of defining and building software products. There are at least five distinct models for organizing the software quality management function:

1. Developers do their own QA.
2. Test or QA engineers are integrated within the development teams.
3. A separate QA group belongs to the engineering manager or VP.
4. A separate QA group belongs to a VP other than the engineering VP.
5. A separate QA organization reports directly to the senior business manager (or a VP of quality who then reports to him or her).

The variety of specific solutions is not surprising, because the industry is still struggling to figure out this problem. Like the software business in general, each company seems intent on inventing its own model for software quality management. Because all the models are based on a do-it-yourself approach, they are subject to the problems identified earlier. Outsourcing software QA activities is an emerging model that offers the business manager a viable option to solving product quality and quality management problems.

Historically, QA outsourcing consisted of low-cost, fast-turnaround supplements to internal testing efforts. Several outsourcing companies thrived by providing compatibility testing of software against various hardware platforms and components.

Typically, client software companies would be running late on development and lack the in-house resources or equipment for fast-turnaround compatibility testing. So they turned to software QA outsourcing, contracting with independent test labs for specific test projects. And while this offered independence and objectivity, it aimed at solving a QA manager’s staffing shortfall, rather than a business manager’s basic quality management problem.

This early model of outsourcing testing is rapidly evolving as major companies try to improve their quality processes. The use of outsourcing is not only accelerating, but changing, as is illustrated by an outsourcing relationship with a leading PC manufacturer.

In 1995, the PC manufacturer started systematically investigating testing laboratories, which it then used on small, noncritical projects that were not adequately staffed internally. There were reviews after each
early project that tested localized software versions. The reviews identified how to improve the testing and communications processes on the next project. Thus, over time, the manufacturer developed trained, trusted people available to the its test organization for overflow work. The organization also planned to outsource some portion of the work and develop a set of trusted, long-term vendors.

By 1997, the manufacturer had decided not to grow its internal testing resources at the rate necessary to deal with an exploding workload. Instead, it formed an internal group whose sole function was management of software test outsourcing activities. A key strategy was to encourage the best vendors to open local labs to improve focus and communications.

In early 1998, QualityLogic, Inc. opened a dedicated lab as a joint venture with another company near the manufacturer's facilities. This lab marked a watershed for the test outsourcing industry in two critical ways. First, it was the first instance of a local software testing lab dedicated to working with a single customer at that customer's invitation. Second, the lab was entirely staffed by local people, many of whom the manufacturer had employed as software QA engineers. The new lab manager, who formerly headed the manufacturer’s test center, brought with him a number of senior software test engineers.

A further evolution is already in process, whereby companies are completely outsourcing some or all aspects of the software quality management function. For example, several organizations have engaged QualityLogic to build and manage their entire software quality function. The vendor hires the company’s existing staff or new staff members, as required, who then become an integral part of the client organization.

The team works on the client site, reporting directly to the business manager or through a designated representative. The vendor’s QA manager is responsible to the business manager for ensuring product and process quality within the defined budget. In fact, the vendor’s QA manager is also the client’s business manager for the specific software QA activity involved.

In all cases, the vendor has a direct company-to-company business relationship with the business manager. In other words, the vendor is solving the business manager’s problem at the same time as it solves the engineering organization’s quality control problems.

This model opens the door for the outsourced QA organization to be an influential participant in the client’s internal development process and tool improvement initiatives. The vendor not only conducts the actual testing activities, but also provides the clients with quality assurance services. The activities include implementing both a defect tracking and a configuration management process (and tools), as well as planning and implementing other process improvement actions.
THE FUTURE OF SOFTWARE QUALITY MANAGEMENT

In determining the future management of the software quality function, early successes indicate that the next logical development is outsourcing the entire QA function, or some appropriate portion thereof. This outsourcing model can directly address the critical cultural and management problems identified in this article. It can also provide improved quality and cost savings for the software company served.

These advantages result from the unique characteristics of the outsourced QA team. First, many of the cultural problems are solved, because the personnel belong to a company whose primary focus is software QA. In such an organization, the software QA engineer is a “first-class” citizen, with all of the status and advantages the term implies. There is a well-defined career path, with the associated training and financial rewards. Stability and maturity can develop because the QA engineers are motivated to stay with the organization and develop as first-rate professionals.

Second, the QA team is set up as a profit-and-loss center with its own competent P&L or business manager (who is the vendor’s QA manager). Therefore, the team has a profit motive for doing a better and more efficient job of providing the customer with software QA services.

Although top-notch internal QA teams are often dedicated and self-sacrificing, it is extremely difficult for a company to financially reward them when they do a great job. QA is not a typical career path to senior management positions, and QA salary levels are generally capped below those of development. Even when a company offers a bonus plan or stock options, such rewards are only indirectly tied to the actual effectiveness and efficiency of the QA team.

By contrast, when a QA team is set up as its own P&L center, it has a very tangible financial motivation for finding the most efficient ways to be most effective at its tasks. While an internal QA manager has little incentive to terminate a “temp” when the project is complete, a P&L manager with a bonus tied to financial results does have this incentive. When equipment is no longer required to perform a testing task, the internal QA group typically keeps it for some undefined future use. A P&L manager cannot afford to keep unproductive equipment as an expense. Most importantly, a profit-motivated group with an experienced management team will find creative ways to increase effectiveness, making the customer happy, and improve the efficiency of the activities — that is, decrease costs.

Dozens of QA organizations waste thousands of dollars and hours of time attempting to automate testing — only to fail. Not only did the team lack the experience required to succeed, but there was no serious enough consequence for failure. Neither factor operates in an outsourced QA team. The costs of failure are reflected in the team’s paychecks, and the relationship with their single customer is placed at significant risk. A
broken promise to automate testing can cause serious mistrust, ending in potential disaster for both the client company and the outsourced QA team.

The third critical factor is the direct relationship between the outsourced QA team and the business manager of their “parent” company (i.e., the customer that the QA team came from). This alone solves both critical problems of software business managers. The very act of making the QA team independent and directly responsible to the business manager (instead of an engineering or other vice-president) places strategic emphasis on software QA. In addition, the business manager has an effective mechanism for monitoring the quality of products under development, in order to take decisive actions.

Through its direct relationship with the business manager, the QA team can also influence the overall software development process. The relationship offers power to “push back” development managers and teams who are shortcutting their own processes. This cannot happen effectively when QA reports to the same vice-president as development.

The QA team can also suggest improvements to the development process that will enhance product quality and increase effectiveness. For example, programming hooks can be added to support test automation, or the product architecture standards can be improved to enhance testability and maintenance.

Finally, outsourcing software QA can result in lowered overall costs for the client company. These take the form of improved quality and lower costs for customer support, of interim fixes and releases, and of better customer retention. In addition, because a profit-oriented QA team is more cost conscious than an internal team, the software QA organization’s cost savings can be passed along to the client. Finally, in the new model of full QA function outsourcing, costs can be lowered even more, as there is more emphasis on process improvement for the entire development cycle.

James Mater is executive vice-president and general manager of the services division of QualityLogic, Inc. He was founder and CEO of Revision Labs, Inc., a leading software QA outsourcing company that recently merged with Genoa Technology to form QualityLogic. Before getting involved in the software test industry, Mater held product management positions at Tektronix and Floating Point Systems.

NOTES
1. The term P&L manager refers to the executive ultimately responsible for both the revenue and expenses for the product organization. In larger companies, this is likely to be a division general manager or president. In smaller companies, it is likely to be the CEO or president. In this article “business manager” will be substituted for “P&L manager” in most cases, as the former term is more commonly used.

2. Product quality management consists of the quality management function (ensuring that good quality policies are in place and enforced), the quality assurance function (devel-
oping and implementing practices and processes that ensure that quality products are produced, and the quality control function (actual testing of products to ensure conformance to customer requirements).

3. Most organizations call these criteria “requirements.” These are the specifications that the organization believes a product must meet in order to satisfy a customer need.

4. Professionally” means that the team provides information in a form, at a time, and in a way that is perceived as non-threatening, objective, and valuable. There is no appearance of a hidden bias or agenda. In short, the test team is respected and listened to by all parties. This is not usually the case with test teams.

5. While a number of companies have contracted to put dedicated software test teams on a customer’s site, these companies have typically not been dedicated software testing companies, nor have they put dedicated labs in place without specific long-term contracts.