IBM Software
Rational

Thought Leadership White Paper

Value-driven quality management for complex systems

Six strategies for reducing cost and risk

Introduction

Consider what software means to your business today. Is it just another component in a larger product or system? Or is it fast becoming *the* differentiator that separates your products in a saturated marketplace? The reality is that companies across many industries are increasingly relying on software to innovate and create the smarter products and systems that customers and marketplaces are demanding. For companies whose business outcomes hinge on software working correctly, quality management is vital to success. At the same time, software quality efforts, which require more rigorous processes and tools to manage issues such as compliance and traceability, traditionally have been seen as add-on costs of development activities. As software's complexity increases and its importance grows, this traditional mindset and status quo approach to quality can lead to serious problems.

Given the danger that software failures can pose to budgets, brands and even people, it is critical to get quality right. The challenge is that quality management is a tricky balancing act that must factor in time, costs and risks. Get it wrong, and you could face issues ranging from unsustainable costs, missed windows of opportunity and unhappy customers to a massive recall or the complete failure of a system at a critical moment, potentially resulting in loss of life or a failed mission. Get quality management right, however, and you can achieve a positive operational return on investment (ROI) from efficiencies gained in development activities. But that's only the beginning of the potential benefits. With effective quality management, you can also create opportunities to deliver critical but difficult-to-qualify benefits, such as improved market share, higher customer satisfaction and increased brand equity

This paper examines the ROI associated with quality management best practices. While each of these practices should be considered in its own right, collectively, they contribute to an even stronger business case for applying quality management as a solution that not only increases quality but also reduces the cost of quality. As this paper will demonstrate, quality management best practices do contribute to process improvements, thus improving quality while reducing cost, allowing you to have your cake and eat it too.

What's the problem with quality?

Complexity is now a given in many products and systems. Software, microelectronics, sensors and mechanical technologies are often combined to create products that can respond to changes; measure conditions; and interact with other products, people and IT systems in new ways.

Consider the potential effect of quality issues: In many industries, large sums of money, the entire success of organizations—and in some cases lives—can hinge on constantly getting complex things right. Yet the success and brand image cultivated over many years can evaporate quickly when something goes wrong. Examples constantly pop up in the news: An aerospace agency faced a roughly US$1 billion loss when a rocket self-destructed because of a bug in a guidance system. In the healthcare industry, software problems and poor quality control in cancer treatment systems designed to deliver precise radiation doses have led to tragic consequences for a number of patients. In these types of cases, what may seem to be a small defect is actually critical to the business and its mission.

One of the biggest challenges related to quality management is how to invest intelligently to minimize risk, given the economic constraints of the business. For example, a consumer products company cannot afford to miss a marketing window of opportunity to fix something that customers will never notice. At the same time, an automotive company cannot afford to ignore an issue that could lead to a massive recall. Figuring out how to relate quality to business outcomes and what constitutes the right level of quality for individual products, however, is not always clear.

Poor business predictability

Depending on the context, quality has different meanings. A customer may consider quality as fitness for use, a manufacturer may define quality as conformance to requirements and a company using a value-driven approach may define quality as a degree of excellence at a certain price point. In each case, however, a quality failure impairs business predictability and may manifest itself in one or more of the following ways:

- Operational difficulties—Poor quality can affect the developmental and operational aspects of product life cycles, ultimately derailing deadlines and driving higher project costs. For example, during development, constant replanning may be required as defects and poor alignment with requirements are discovered, leading to late-cycle rework.
- Through-life costs—Recalls, updates, warranty claims and litigation are all potential costs of prioritizing delivery timing over product quality. By imposing testing cycle cutoffs to meet delivery schedules rather than considering quality, you inevitably move unpredictability into the operational space, where the effect can be much more severe.
- Long-term business value decline—When customers are directly affected by quality issues, the long-term costs, such as loss of market share and brand equity, can be very high.

Improving the delivery process

So how can you change the delivery process to address quality problems early in the life cycle rather than masking their effects until a later stage? The safest way is to use proven approaches and process frameworks, such as the Capability Maturity Model, Capability Maturity Model Integration (CMMI), agile, clean room or other general or domain-specific approaches. These approaches are well documented, so this paper does not delve into them. Rather, it discusses best practices you can apply regardless of your chosen development approach or process framework.

A quality improvement program is an investment, and, as such, you need to clearly understand the benefits and return on investment to manage expectations appropriately. A simple definition of ROI is:

ROI = (cost saved - investment)/investment

Focusing on best practices that can deliver a positive return provides a way to separate practices that will deliver tangible business results from merely interesting ideas. Moreover, this paper will explore how you can further justify investments by potentially delivering savings that are bigger than the sum of the individual returns.

Software delivery strategies

During any software delivery process, “When do we release?” is a key question with no single “correct” answer. Rather you must consider project-specific variables, such as the cost of delays, the opportunity value of early delivery, marketplace quality expectations and the costs associated with defects. Ultimately, the delivery strategy will be based on the actual or perceived importance of each variable. Typically, software delivery strategies are schedule driven, quality driven or risk driven.

The pitfalls of a schedule-driven strategy

Schedule-driven delivery effectively implies “deliver on time, regardless of other factors” and is often used in consumer environments to hit marketplace windows or in contractually governed environments, where delay penalties may apply. Such a strategy can prove profitable in the short term, but it also introduces a high potential for quality risks and relies on customers as unwitting testers. Quality issues are often magnified, given that software contractors frequently get paid on a time and materials basis regardless of the quality of software they deliver. In many cases, you may even end up paying extra for them to fix their own defects, so the potential costs of defects can add up quickly.

According to the Carnegie Mellon Software Engineering Institute, “Data indicate that 60–80 percent of the cost of software development is in rework.”[1] Potential costs don’t stop there—litigation is an increasingly significant cost risk factor. Moreover, increasing costs and loss of business affect the ability to innovate—leading to a vicious circle of problems.

The shortcomings of a quality-driven strategy

Quality-driven delivery can also be costly but for different reasons. The release timing for this approach is governed by achieving the right quality—but how do you define that? Achieving zero defects is practically impossible, given that there is no way to determine how many defects still exist in a piece of code or the probability of detecting those defects in use. A target based on defects fixed might be more realistic—but it’s still impossible to know the number of remaining defects in the product. Either way, you may be wasting valuable time and money on issues that aren’t significant to product success.

The advantages of a risk-driven strategy

In terms of cost-effectively balancing quality risk versus time-to-market considerations, a risk-driven approach may be ideal. A risk-driven strategy is a refinement of a quality-driven approach that optimizes risk exposure against development cost or time. The overall risk exposure is the sum of the exposures for all identified risks associated with the project:

Risk exposure = $\sum$ (probability of loss x size of loss)

The optimal time to release is when the total risk exposure is minimal, typically around the time where the risk associated with competitive threats starts to outweigh the risk reduction associated with further quality improvements, as illustrated in figure 1.

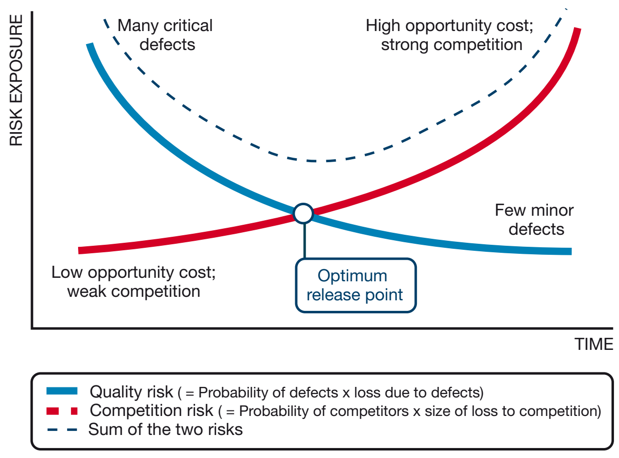

Figure 1: Balancing risk exposure and opportunity cost

With different types of products, the optimal time to release can vary widely and is a tough balancing act. Move too soon, and customers may be upset with quality issues. Wait too long, and your competition will beat you to the punch and dominate the space. So, in the end, the optimal time to release lies where your overall risk is minimal. Consider the following examples.

- Mass-market products—In the consumer electronics marketplace, time to market is critical to product success. Here the opportunity cost of delaying a release to improve quality can be high, thus pushing the optimum release point to an earlier date.
- Safety-critical applications—In applications such as flight control software, quality may be the most critical parameter, given the potential size of loss caused by a defect. In this case, the optimum release point would be a later date.
- Systems with high-availability requirements—Systems, such as those for mobile communications in the telecommunications industry, have extremely high availability requirements because of the commercial implications of system outages. As a result, the potential loss resulting from a defect makes quality critical, pushing the optimum release point to a later date.

Although development and delivery teams have control over time to release, many external factors in your industry and marketplace dictate your time to market. In practice, the required time to market typically may be before your optimal time to release, where your overall risk exposure is the lowest. Tackling this problem requires approaches to accelerate risk reduction to hit time-to-market windows.

Quality management versus testing

If a faster reduction in risk is the goal, how do you achieve it? The answer is not testing, which is focused on discovering defects rather than on being a risk reduction mechanism in its own right. Quality management, which is the implementation of practices to proactively reduce risk, is the better answer. By choosing quality management practices with the potential to deliver a positive ROI, you can justify risk reduction measures from not only a quality standpoint but also a financial standpoint.

Traditionally, testing has been a late-stage activity in the development life cycle, conducted between the construction of software and its eventual release. Before you can test anything, testable entities must exist, and you can't know the level of risk until testing determines the defect density in the entities being tested.

Quality management can be considered as its own life cycle within the overall software development life cycle, as shown in figure 2. Its activities take place across the development process and are synchronized with the development process in key points. For example, requirements definition provides the input for the start of test planning, software construction provides the entities for test execution, and testing provides the results data for defect management and resolution.

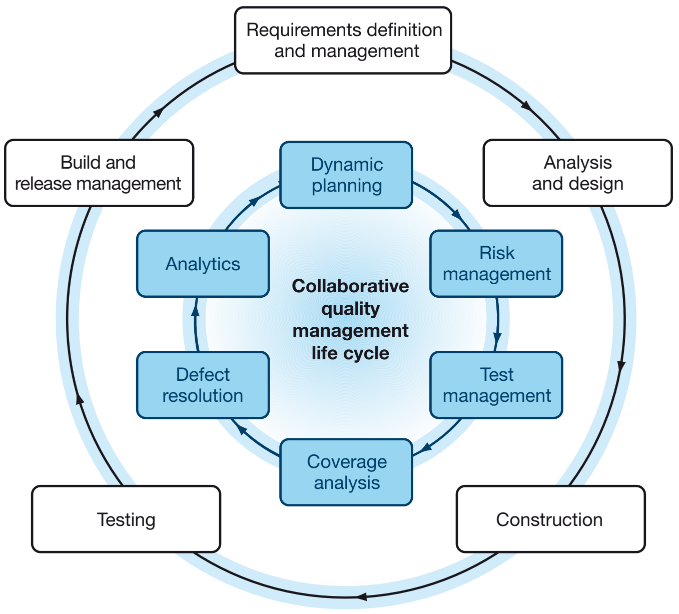

Figure 2: The quality management life cycle within the software development life cycle

The entire process is closely linked to change and configuration management. Discovered defects are work items (or change requests) that must be managed along with other change types in the change management process.

There are many strategies for reducing risk, so it is essential to distinguish between those that are good ideas and those that also deliver a positive ROI. The following are six strategies that are proven to deliver a positive ROI.

Strategy 1: Drive testing from requirements

Linking test definitions to requirements can help ensure that what you ultimately test reflects requirements. This approach can help reduce risk by helping to ensure that tests demonstrate compliance with requirements. It also reduces the risk of overengineering the product by effectively detecting feature creep in the form of code that can't be linked to any requirement—a factor that is often the source of major quality and instability issues.

The tooling used to maintain the traceability between requirements and tests can have a major effect on productivity. For example, in many cases, teams maintain requirements and test cases in conventional office applications, such as text documents and spreadsheets. Consider a hypothetical medium-size project with some 5,000 requirements and 10,000 test cases. Assuming it takes 20 minutes to locate and link the appropriate artifacts for each requirement, it would take approximately 10 person-months to create the traceability between requirements and test cases.

You could potentially reduce this time to one to two minutes per requirement—and a total of just 10 to 20 person-days—using a dedicated quality management solution with support for capturing traceability links between requirements and test cases. At a nominal hourly rate of US$50, this single change corresponds to a potential saving of around US$75,000.

Strategy 2: Reduce planning overhead with collaborative quality management

Quality management is an activity with implications across the development life cycle. As such, managing the testing plan is a collaborative activity involving many stakeholders. It requires a central repository where stakeholders can share information and access a "single version of the truth." Definition and management of workflows, so that all stakeholders understand expectations placed on them and others, are also important.

How efficiently and effectively stakeholders collaborate can have a big influence on productivity. For example, client interviews conducted by IBM have shown that testers typically only spend 60 percent of their time testing; the rest is spent in collaborative activities such as communicating with engineers, tracking decisions and retrieving information. IBM clients that have automated various collaborative tasks have seen collaboration efficiency improve by 20 percent on average, leading to a roughly eight percent improvement in overall tester productivity. For a team of 100 testers, this could lead to savings of around US$750,000 over 12 months (assuming a US$50 per hour rate). This means a team of 92 testers could do the work of 100, leaving the other 8 to accelerate the delivery schedule.

Strategy 3: Prioritize testing according to risk

Although you cannot completely remove risk from a development program, you can measure it and manage it by taking proportional mitigation actions. As previously discussed, the risk associated with a particular outcome is the product of the probability of the outcome and the size of the resulting loss. You can use this quantification of risk to prioritize testing, helping to ensure that you test high-risk functionality and requirements first. This approach helps make sure that you are optimally using finite testing resources to reduce risk as rapidly as possible in the development cycle.

Tools that automate the measurement, assessment and prioritization of risk provide a way to help optimize a risk-driven development workflow. For example, if your target is to test for 99 percent of identified risk, without a risk-based approach, you would need to execute 99 percent of tests. By measuring risk and prioritizing tests accordingly, you may be able to test for 99 percent of identified risk with, for example, 90 percent of tests—effectively improving testing productivity by 10 percent. Again, for a team of 100 testers, this can correspond to savings in excess of US$900,000 over 12 months (assuming a US$50 per hour rate).

Strategy 4: Integrate testing and quality management to improve accuracy and repeatability

Integrating both manual and automated testing within the quality management environment can deliver several benefits, including:

- Improved management of tests
- A greater likelihood of executing the correct set of tests
- Enhanced collaboration resulting from better information on testing status and results

Automation delivers further advantages because it increases the likelihood of executing the tests—resulting in earlier defect detection and more consistent regression testing. Although the benefits of testing integration are clear, a quantifiable business case depends on the nature and frequency of the tests being automated and on the costs to implement the automation.

Strategy 5: Automate reporting to improve efficiency, consistency and decision making

Reporting is a key element of the quality management process because it facilitates educated decision making based on the information captured across the process. The costs of reporting correspond to the complexity of reports and how many people are required to locate and format the information as well as how often you generate reports. Manual reporting costs tend to be proportional to the number of report versions created. By automating the process, however, you can produce subsequent reports with minimal additional resources and cost.

Automation also helps improve the accuracy, consistency and timeliness of reporting—factors that can positively influence project management, productivity and quality, all of which have implications for project ROI. A specific business case depends on the complexity and number of reports you typically generate; however, given that many reports require frequent updates, it is likely that automation will show positive ROI.

Strategy 6: Improve efficiency by applying smart defect management to eliminate duplicates

As project complexity increases, the number of defects detected can be expected to increase—but often defects are reported more than once. An IBM study identified this problem as a significant hidden cost factor in development projects.

Consider a project in which testers detect 1,000 defects, all of which developers must investigate. Suppose that 20 percent of those defects are duplicates and that each duplicate requires an average of two hours of development time to establish that it has already been fixed. This corresponds to 400 hours of developer time—or approximately 50 days (equaling about US$20,000) of overhead simply dealing with duplicates—an activity that adds no incremental value to the project.

In reality, a defect actually may be detected many times. Moreover, in a geographically distributed team, which is typical of today's complex projects, it may take significantly longer than two hours of developer time to detect the duplication. Tooling that automates the detection of duplicate defects as testers log them can therefore help reduce overhead costs and shorten delivery timelines.

The overall effect of risk reduction strategies

Each of the individual strategies discussed in this paper delivers clear benefits and is worth considering in its own right. However, combining all of the strategies to implement quality management as a life cycle can contribute to a much larger ROI. In other words, the overall return is bigger than the sum of all of the individual returns. That's because a thorough quality management life cycle can root out defects earlier—when they are cheaper to fix.

CMMI for process maturity provides a good example for analyzing the overall effect of a quality management life cycle because it can be related to a lot of industry and academic data. Figure 3 shows a detailed IBM analysis of comprehensive data from both industry and academic studies. The chart demonstrates that the transition from one CMMI level to the next can have a significant effect on the efficiency of the quality process. For example, one effect is a decrease in the ratio of detected defects to escaped defects during functional testing as the CMMI level achieved increases. Figure 3 shows the relationship among three measures at each CMMI level.

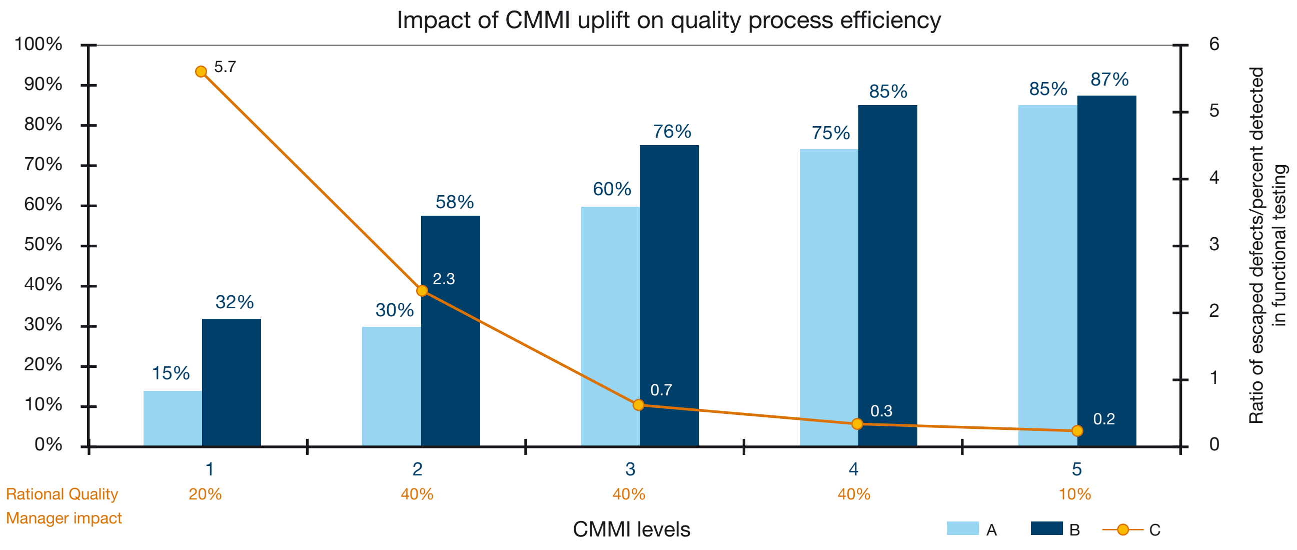

Figure 3: This figure shows the relationship among three measures at each CMMI level. The bars represented by A show initial detected defects as a percentage of total defects. The bars represented by B show detected defects as a percentage of total defects with the best practices deployed. Line C shows the initial ratio of escaping defects to detected defects.

For illustrative purposes, consider a project that is at level two, where functional testing is detecting 1,000 defects. Line C in figure 3 suggests that testing will miss 2,300 defects, resulting in a total of 3,300. With the help of best practices, defect detection rates could be increased to 58 percent, resulting in the detection of 914 more defects.

Fixing defects earlier for potentially significant cost savings

The later you discover defects in the development cycle, the more you pay: An IBM Global Business Services study found that it costs 7 to 14 times more to fix a defect in user acceptance than during unit testing. Using the lower of the two figures, if it costs approximately US$120 to fix a defect during unit testing, then by finding the 914 additional defects earlier, an organization could save the following:

US$120 x 914 x (7 – 1) = US$658,080

Compounding benefits: Greater than the sum of the parts

The various quality management strategies outlined in this paper can all deliver efficiency or productivity benefits. So when you take those benefits into account, along with the fact that teams will detect more than 1,000 defects for many large projects, and consider the potential sum of risk reduction strategy savings plus early defect detection savings, you arrive at potentially very large operational cost savings. And this doesn't even take into account downstream benefits related to protecting the brand image and customer satisfaction with higher-quality offerings.

Improving quality with the help of IBM solutions

Achieving the benefits of thorough quality life cycle management requires a strategic combination of integrated capabilities that can help you manage collaboration and testing. IBM offers a number of domain-focused workbenches to address the needs of different software and systems development teams. The IBM Rational® Workbench for Systems and Software Engineering is designed to help you deliver high quality systems while reducing costs and risks. The Workbench, which comprises IBM Rational DOORS® software, IBM Rational Rhapsody® software, IBM Rational Quality Manager software and IBM Rational Team Concert™ software together with best practice process offerings and deployment services, provides a core development capability, encompassing requirements management, model-driven development, quality management, collaboration and workflow, and change and configuration management. Rational Workbench for Systems and Software Engineering capabilities may be extended through integrations with other IBM and third-party offerings.

Providing a quality management hub

Rational Quality Manager software provides a collaborative, customizable web-based test and quality management hub for the quality management life cycle. Whether they are down the hall or across the globe, quality professionals and other decision makers can use Rational Quality Manager software to collaborate on virtually all aspects of quality maturation, such as test planning and management, risk-based testing, and defect management, including duplicate defect detection.

The solution uses a Web 2.0–style interface and flexible, automated reporting capabilities to provide project members up-to-the-minute project metrics and analytics customized to their role, so they can detect defects earlier and keep projects on track. Decision makers can also use detailed, timely information to identify trends and make ongoing improvements.

Delivering quality in complex systems is a wide-ranging activity that requires interoperation with other technologies and tooling. Rational Quality Manager software can be integrated with other Rational offerings and also provides open interfaces that you can use to connect to other testing solutions from both IBM and third-party vendors.

Supporting requirements-driven testing

Traceability is essential to tracking whether a product ultimately meets requirements. To support this need, Rational Quality Manager software is integrated with Rational DOORS software, which provides comprehensive features for structuring, managing, tracking and tracing requirements from business needs, through technical specifications, to software and system test cases.

Supporting model-driven testing

The increasing complexity of systems has caused a shift to model-based development techniques for many delivery teams, allowing greater team productivity and better understanding and communication of design information through the use of visual modeling languages such as the Systems Modeling Language (SysML) and the Unified Modeling Language (UML). Model-driven testing enables the testing activity to keep up with model-driven design productivity. Rational Quality Manager software is integrated with IBM Rational Rhapsody TestConductor Add On software, allowing model-based testing to be efficiently managed as part of the quality management process.

Unifying defect management and change workflows

Constant change is the norm in complex systems development as requirements evolve and you detect and resolve errors and defects. As a result, change management efficiency can significantly influence project success. Rational Quality Management software is integrated with Rational Team Concert software to unify the defect and change management workflows. The integration also enables you to automate notifications of build status from Rational Team Concert software to Rational Quality Manager software, so you can execute tests against new work products and manage the resolution of defects found in testing in a single change management environment.

Conclusion

The momentum of technology, coupled with customer demands, is pushing organizations to deliver ever-smarter products. Given the added complexity in development processes, finding ways to continually optimize time, cost and quality is critical to ongoing successful innovation. A risk-driven development approach not only can help you balance quality risk with time-to-market drivers to better support business needs but also can do so while delivering a positive ROI. The cost benefits are achieved through addressing defects earlier in the development process when they are less expensive to fix as well as through productivity and efficiency gains in development processes.

What's more, a positive ROI and improved project outcomes are only the start of the benefits from a risk-based approach to quality management. Such an approach can also support long-term benefits that cannot be easily quantified but are key to long-term business growth and success, including customer satisfaction, reduced in-service costs, and a strong reputation and brand. When you are ready to improve the quality of your systems while reducing risks and costs, look no further than IBM for the tools needed to implement the strategies outlined in this paper.

For more information

To learn more about IBM products that support quality management, please contact your IBM marketing representative or IBM Business Partner, or visit: **ibm.com**/software/rational/offerings/quality

Additionally, financing solutions from IBM Global Financing can enable effective cash management, protection from technology obsolescence, improved total cost of ownership and return on investment. Also, our Global Asset Recovery Services help address environmental concerns with new, more energy-efficient solutions. For more information on IBM Global Financing, visit: **ibm.com**/financing

[1] Paul D. Nielsen, “About Us: From Director and CEO Paul D. Nielsen,” Carnegie Mellon Software Engineering Institute, http://www.sei.cmu.edu/about/message

Rational. software