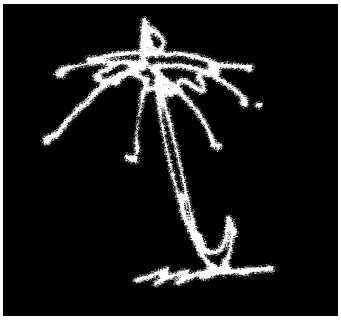


Managing Risk in Software Maintenance

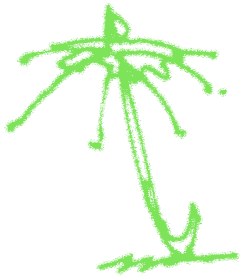
ROBERT N. CHARETTE, ITABHI
KEVIN MACG. ADAMS, QED Systems
MARY B. WHITE, Andrulis Research

Risk management in maintenance differs in major ways from risk management in development. Risk opportunities are more frequent, risks come from more diverse sources, and projects have less freedom to act on them. The authors describe how they dealt with these differences in a large US Navy software maintenance organization.

Much of the literature about software risk management pertains specifically to the development of new software. Software maintenance has with a few notable exceptions[1-3] been overlooked. This is somewhat understandable given the ugly duckling image that maintenance has in the software engineering community. On the other hand, the enormous sums spent on software today are typically not going to development, but to this "unimportant" side of software engineering. And for good reason: it is not unusual for a software application to be used for two decades or more.

Correcting this oversight is not a simple matter of transferring development-oriented risk management practices to maintenance organizations. Of course, the two areas share certain risk factors—requirements volatility, staffing inexperience, and tight time constraints—but there are also significant differences. Risk management in maintenance generally provides more opportunities for risk, along with less freedom to mitigate them. Maintainers must work with an existing system—sometimes decades old—with little or no documentation. The system often has layers of changes atop the original design and interfaces with myriad systems that the developers never even conceived of. Any change to this intricate web can cause deadly side effects.

0740-7459/97/$10.00 © 1997 IEEE



Risk management in maintenance also involves more attention to customer-related issues. A maintained system has an established user group that relies on its smooth operation. This contrasts sharply to, say, alpha and beta site evaluation of a new release, in which the user group is just beginning to connect with the system. Typically these users are also aware of the risks, have adjusted their expectations, and tend to be more tolerant of erratic operation. For these reasons, a maintenance risk carries a higher risk consequence (loss of an established user base) than a similar risk in development.

The application of risk management in a maintenance environment must be sensitive to existing risks.

Overall, the application of risk management in a maintenance environment must be much more sensitive to existing risks and how they will be mitigated. In this article, we describe how we addressed the need for increased sensitivity in applying risk management within the US Navy Maintenance Support Office. We hope our experiences with NMSO spark others to investigate this forgotten area of risk management.

NMSO MISSION AREAS

The NMSO provides the US Navy and other Department of Defense logistic activities with business process improvements, planning products, and information technology to support the timely and high-quality maintenance of military platforms. NMSO has four primary mission areas:

♦ *Naval shipyard business improvement.* This area encompasses all the business and workflow processes and organizational and information technology issues associated with the core business functions of Naval shipyards. It revolves around the repair, maintenance, overhaul, and modernization of US Navy combat ships and their attendant weapons, propulsion, and auxiliary systems. NMSO supports the central business processes that the Navy's four shipyards use in conducting ship-related work. It models every transaction from contract award to final certification and then links them to specific software modules in the BAIM (Baseline Advanced Industrial Management) software system.

♦ *Joint service maintenance initiatives.* This mission area supports standard DoD project management processes and their supporting information technology. NMSO acts as the agent for the Joint Logistics Systems Center, whose role is to maintain and modify these processes and software for use in the broader DoD arena. BAIM is the DoD choice for use on large projects in all DoD depots.

♦ *Fleet maintenance initiatives.* This area continues to evolve as the Navy reengineers the concept of ship maintenance for the next century. The Navy is establishing centers based on "regional maintenance." The idea is to transform the Navy's stovepipe maintenance processes into a more homogeneous form that supports a fleet's ships right in their home ports instead of periodically performing long, complex overhauls in the shipyards. In this way, regional maintenance becomes part of a continuous-maintenance approach. The role of the centers is to supply assets to support the fleet in port. NMSO's role is to provide common, reusable planning products and supporting information technology for implementing regional maintenance.

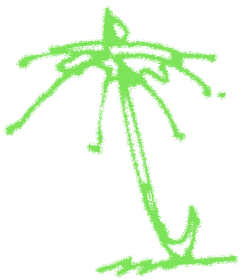
♦ *Joint/Navy Reengineering Center of Excellence.* The Center gathers, documents and disseminates lessons NMSO has learned in reengineering DoD and Navy maintenance efforts to support planned and future DoD process improvements. The NMSO engineering staff transfers the lessons learned to a range of DoD customers by supporting reengineering in modeling, simulation, computer-aided software engineering, and the reverse engineering of legacy code.

As these mission areas illustrate, NMSO is a somewhat atypical software maintenance shop because it must also perform a range of roles that are not strictly software maintenance. However, providing software and system enhancements and corrections to fielded software systems is NMSO's most resource-intensive task, so we believe the approaches described here can apply to more typical software maintenance organizations—at least to some degree.

PILOT PROJECT

In 1994, as part of a business reengineering effort, NMSO management created a process improvement activity aimed at upgrading the overall quality of internal software maintenance. One area they wanted to improve was their ability to manage software-related project risks. As part of this work, they used a pilot software development project sponsored by the DoD—the Integrated CASE Project Management System. A goal of the project was to get more experience on how to apply risk management across NMSO by seeing the results of applying it to a development project. NMSO management selected the I-CASE PMS project because they expected to see many opportunities for risk, both internal and external, over the project's 18-month life.

Our task was to work with NMSO staff to devise a project management plan that included a formal risk management process. Because the pilot was a development, not a maintenance, project we decided to use a risk management approach NMSO had already defined. The approach is based on a simplified version of Charette's[4] seven steps of risk management: identify, estimate, evaluate,



plan, obtain resources, control, and monitor. The identification step is augmented by the Software Engineering Institute's software risk taxonomy.[5]

A significant part of the NMSO approach is its simple consensus method of risk assessment and mitigation. Within each of these primary phases are several support activities, such as estimating the likelihood and consequences of a risk, prioritizing it, and recommending mitigation approaches.

Risk assessment. With the help of the SEI, NMSO created a risk baseline at the start of the project using the SEI's software risk evaluation process.[6] NMSO uses the baseline to periodically identify risks using the SEI taxonomy. During the life of the project, we identified 168 risks, with requirement-related risks occurring most frequently. The team rated each risk as to its "perceived severity" and "likelihood of occurrence" using a simple low-medium-high scale for a risk's consequences and an improbable-probable-certain scale for its likelihood. Each scale factor had a standard definition to help maintain estimation consistency. This information yielded an expected value, which let the team rank each risk from most significant to least significant. The team members and the project manager then reviewed the list of all identified, prioritized risks, selecting the top five for risk mitigation. (We restricted mitigation to the top five because resources were limited. We recognize that this was itself a "risky" strategy, but it was the most practical approach given the available resources.)

Risk mitigation. With the top five risks in hand, we devised a mitigation plan for each one. Within the plan we listed the actions needed and their completion dates, a contingency plan, and the person responsible for ensuring that the mitigation plan was properly executed. We then placed the risks and their mitigation plans on a risk status tracking list, which was available for all project members to review periodically. The person responsible for the list provided a weekly update.

We reassessed risks every six to eight weeks, usually in parallel with major project deliverables. Reviews of risk mitigation status, on the other hand, were done every week. All project members actively participated in risk management. Data from these assessments made it easier to strategically manage risks for the project as a whole, as well as to mitigate individual risks.[7]

Lessons learned. Overall, this approach proved successful. The NMSO-defined risk management approach was basically sound and seemed easy to understand. The simple method and consensus approach to assessing and mitigating risks kept everyone informed as to where the difficult areas were and what we needed to do to overcome them, both as individuals and as a group. It also kept us focused on addressing the real risks, not the perceived ones.

The project team felt that proactively assessing and mitigating the risks not only increased the probability of project success but increased individual ownership and made the project more fun to work on. We also learned that if we assume the project is risky, it is easier to see the need for a risk management process from the start.

Because everyone participated in identifying and mitigating risks, there were fewer inhibitions about dealing with them later—something many software projects fail to overcome. By using a standard process, the team was able to communicate serious risks to upper management without feeling personally threatened or needing to place blame.

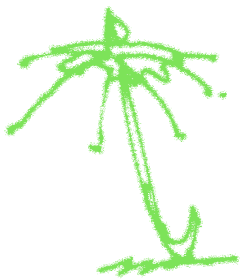
ORGANIZATION-WIDE GOALS

Like most highly visible pilot projects the I-CASE PMS project was not your typical development project. We had complete upper management support for what we were doing, fully qualified and aggressive project management and staff for the life of the project, and very little personnel turnover. Everyone received focused risk management training, and we had stable funding.

In attempting to institutionalize risk management within the larger NMSO culture we faced a far different environment: skeptical staff and management, high personnel turnover, multiple projects with competing claims to resources, diverse and very vocal customers, and erratic funding. Moreover, our focus now had to be on the needs of a maintenance organization, not on the needs of a development project, which was our focus on the I-CASE PMS project. We guessed we would have to modify the risk management process we had used—we just didn't know by how much. We were certain of one thing: how we applied risk management in our maintenance efforts was going to be different from how we did it during the I-CASE PMS development.

Given this new environment, we thought it wise to follow a three-step change-management approach, one that was sensitive to cultural reasons for resisting change.[8] We decided to create a risk management infrastructure as the first step. The second step was to establish implementation support. The third and last step was to devise a system for monitoring the progress of institutionalizing risk management and for getting feedback to improve the risk management process.

Proactively assessing and mitigating risks increased the probablility of project success and made the project more fun.



RISK TERMINOLOGY: A BASIS FOR CONSENSUS

We know that in NMSO and Rockwell, and we suspect in most other organizations as well, "risk," "problem," "concern," "worry," and "issue" tend to be used interchangeably. This increases the chance for misunderstandings. Organizations must take care to at least use "risk" and "problem" precisely and to let others know what is meant by "issue," "concern," or "worry."

At NMSO, we made it clear to staff that a *risk* is an event with a likelihood of occurrence and some potentially negative consequence. Put more simply: a risk is a potential problem; a problem is a risk whose likelihood has reached 100 percent. It is a certainty. When respondents did see risks as separate from problems, they generally saw them as associated with some future decision: "Risks are about the future, light-years away. Problems are now."

It was much harder to nail down what people meant by "concern," "worry," or "issue." To some at NMSO, an "issue" was an event with a greater likelihood and consequence than a "concern" or "worry." Others felt that a "concern" or "worry" carried more risk than an "issue." And to still others each term meant a "problem."

To overcome this we did not try to define and enforce an NMSO-standard terminology. Instead we encouraged people to use only "risk" and "problem" in their discussions. The training courses sensitize everyone to the point that when someone inadvertently uses "issue," they are often immediately asked "Do you mean a problem or a risk?" We also found that we needed to make explicit the meanings of words used to describe risk: "possible," "probable," "expected," and "doubtful" were the most common adjectives. Because individuals use these terms interchangeably (and inconsistently) to indicate a risk's likelihood or consequence or both, we train NMSO staff to always describe a risk in terms of its likelihood, consequence, and timing and to include their degree of confidence in each estimate. We also defined standardized but simple scales for each term. For a risk's timing, for example, the scale was *immediate* (one to three months away), *soon* (three to six months away), and *later* (more than six month's away).

These measures, as well as encouraging the staff to ask for terminology clarification, reinforces good communication about risk—a key part of effective risk management.

RISK MANAGEMENT INFRASTRUCTURE

Our goal for the infrastructure activities was to build on the existing positive experience of performing risk management in the organization, as well as to put in place the mechanisms to implement the process in a maintenance context. The creation of the infrastructure involved three tasks: perform a cultural assessment, define success benchmarks, and tailor and enhance the process we used on the I-CASE PMS project.

Cultural assessment. From our experience in introducing other software process improvements into NMSO, we knew that institutionalizing risk management was not going to be a plug-and-play operation. Unlike a new development situation, in which (at least in our experience) new processes can be introduced relatively easily, people already familiar with a set of processes were being asked to add yet another to their already excessive workload. We knew that if we did not take the time to understand the organizational culture, we could inadvertently set up a backlash against risk management. (This is also true for new software development projects that are trying to introduce risk management after development has started and a culture has had time to form.) In particular, we did *not* want to

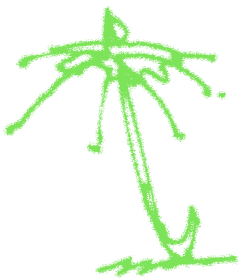
- create a process that would unnecessarily conflict with NMSO's current culture,
- deliver benefits that did not show measurable value directly from using risk management, or
- create unrealistic expectations by selling risk management as a panacea.[9]

We began the cultural assessment by conducting 25 in-depth, structured, and confidential interviews with both NMSO management and line staff. We asked a series of questions to ascertain

- the level of understanding about risk concepts, assessment, and management;
- how individuals perceived and addressed risk within their everyday work environment;
- whether the staff believed a formal or informal risk management process would be effective; and
- what NMSO perceived was the most valued organizational virtue or quality.

The cultural assessment revealed that NMSO had only a rudimentary knowledge of risk and its management. Risk was evidently not a common term used in meetings or general discussions. Respondents used "issue," "concern," "worry," "risk," and "problem" interchangeably. As the sidebar "Risk Terminology: Basis for Consensus" describes, these terms actually have very different meanings. For example, a common response to "Are risks treated separately from problems?" was "I group risks and problems together as issues of concern."

The assessment also revealed certain attitudes about risk. Many believed that using the term was akin to "whining." Others said that "risks were not real," so why bring them up? Others viewed risk as "bad news that required extra work."



On the other hand, nearly everyone agreed that following a disciplined process for risk management would yield benefits. Most said they would use such a process if it were available, but only if

- it was not bureaucratic or time-consuming,
- they could receive sufficient training in how to apply it correctly, and
- the line staff saw management using it too.

The last point is critical. In the pilot project, risk management was "new," visible, had senior management's attention, and operated in an ideal environment. Its success was a high-priority objective. In NMSO's normal operating environment, risk management was now only one among several other "equally important" improvement activities that competed for management's attention and follow-through. Virtually all the line staff saw little benefit in applying a risk management process if managers were not actively involved in mitigating the risks they identified. Also, the line staff needed managers not to consider risk a stigma. These findings are close to those of a similar study performed by Art Gemmer at Rockwell International's Cedar Rapids Collins Division, which is engaged in commercial avionics maintenance and development work similar to NMSO's. [10]

One set of conflicting values came out very clearly in the cultural assessment: Management perceived that product quality was a higher priority than the maintenance release schedule, but line staff felt that the schedule was more important. We would have to resolve this conflict if we were to implement a consistent way to manage risks or problems.

Finally, the assessment showed that NMSO decisions were often accelerating the transformation of risks into problems. NMSO was finding it hard to say no to customers who wanted work that would overly tax their ability to deliver. The reasons were due partly to the Navy's "can do" spirit and partly to the lack of a mechanism to show the customer *why* something could not be done. In short, the assessment revealed a need for some way to assess the amount of risk NMSO deemed acceptable.

We recognize that many of these results are consistent with those from development environments: Conflicts between quality and schedule, a low level of understanding about risk management activities, and resistance to the introduction of risk management. The difference we note is that the culture of a maintenance organization is not as monolithic:

- Maintenance organizations do not have a project completion date to rally around. There is only the next release, and the one after that, and the one after that, ad infinitum.
- Releases are often a mixture of enhancements and corrections for different customers, again reducing the sense of focused purpose.
- Individuals frequently work alone on part of a release and often have sole responsibility for that part. This contrasts with working as part of a team, in which individuals communicate with one another during their work.
- Release planning and personnel responsibilities are not synchronized, which keeps maintenance organizations from attaining a sense of cohesion.

These conditions tend to keep a maintenance organization from forming a single culture, which makes coordination and cooperation on managing risk more difficult and makes the success of risk management much more dependent on an individual's actions.

Success benchmarks. The cultural assessment showed us unmistakably that establishing risk management was going to require much attention to cultural sensitivities. We also saw that we had to demonstrate rapid, visible, and measurable success of the risk management approach if we were to successfully introduce and institutionalize it. We knew this was going to be difficult because it is hard to determine whether a nonoccurring risk results from luck or good risk management. [11]

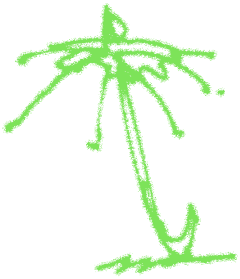
Defining appropriate indicators of success would help us demonstrate that risk management was having a positive effect in NMSO. The benchmarks could also serve as a feedback mechanism to indicate how much progress was being made in introducing risk management into NMSO.

We defined three related categories of benchmarks:

- *Compliance indicators*. We used these as audit indicators to demonstrate if NMSO personnel had received the requisite training before they attempted to apply the process, as well as if risk management was being applied when NMSO policy required. We also used them to quickly pinpoint if the difficulty encountered was caused by the process itself or by a lack of training, and to measure the overall level of risk management institutionalization. The total percentage of NMSO personnel receiving general and/or specialized training is a typical compliance indicator.
- *Effectiveness indicators*. We used these to get a sense of how thorough the risk process is, whether it is noticeably affecting how decisions are being made and how risks are being communicated and managed. The number of unplanned meetings is a typical effectiveness indicator.

Our assessment showed that NMSO decisions were often accelerating the transformation of risks into problems.

- *Value indicators*. Given all the resources being spent on risk management, do the staff, the organization, and, most important, NMSO customers see a noticeable difference? To an individual, for example, the difference may be that they



now have the time or information to do quality work. To the organization, it may be that they can respond to more user requests or hit planned schedules with a quality release. To the customer, it may be more confidence that NMSO will achieve what they promise. Value indicators are measured by conducting periodic surveys of NMSO personnel and customers regarding how their perceptions of risk management positively or negatively affected the software they worked on or received. This trend is assessed and compared with other trends such as the number of trouble reports and customer requests fulfilled. We see value indicators as the most important of the three benchmarks because they are the most direct evidence that the organization is accepting or not accepting institutionalized risk management.

The Year 2000 issue is an example of a risk caused by the lack of information, control, and now time that maintenance faces.

Process tailoring. The goals in this step were to modify the risk process used in the I-CASE PMS project to meet NMSO's culture, work environment, and ability to absorb a new process. On the surface, the process for software development risk management would seem to transfer easily to maintenance. However, although the overall steps of identify, estimate, evaluate, and so on may be the same, the root causes of risk—lack of control, lack of information, and lack of time[1]—are more pronounced in maintenance than in development. There are also more subtle differences, such as what defines success.

Root-cause differences. Of the three root causes, lack of control is probably the most influential. Maintenance is both an event- and interrupt-driven process. Requests for enhancements and corrections flood in. Although many requests can be put off to await a planned release, those that affect critical operations cannot. This makes it extremely difficult to predict maintenance resources. The lack of control is also greater because a software design and architecture exist, and any desired changes are constrained by what was done during development. The nature of maintenance itself exacerbates the lack of control by "consuming" the design with each change. As more and more changes are added, the software becomes increasingly brittle and error-prone. If the software was poorly designed, the lack of control quickly becomes apparent in maintenance.

This consuming process also causes a certain kind of information deficit not found in development. A small change can have a major rippling effect that requires insight into the design to remedy. Development documentation, if it is there at all, is typically little help. Moreover, few developers stay on as maintainers; in fact, organizations tend to put new hires (usually just out of school) into maintenance because it is "not as important" as development. With this attitude, maintenance personnel turnover over is typically high and resources are scarce. All this helps explain why between 20 and 50 percent of all enhancements or defect corrections made during maintenance introduce additional errors.[12]

The lack of time is also greater in maintenance than in development, again because the software is operational. Changes that affect operational effectiveness must be done quickly, regardless of whether they were caused by an error or by a change in the user's operational environment. Demands from multiple users are generally competitive; everyone wants his particular enhancement or fix done at the same time (usually right away). Unlike development, there is no "maintenance phase" for implementing difficult requirements further down the road. The looming Year 2000 issue is an example of a risk caused by the lack of information, control, and now time that maintenance faces.

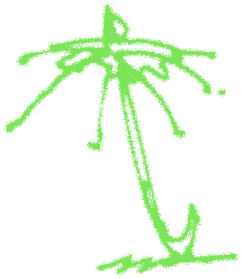
Subtle differences. One difference between development and maintenance that is often overlooked is the definition of success (or failure). In new development, success has a (very) flexible definition. It can mean close to budget, schedule, and meeting requirements. It can also mean the project wasn't scrapped.

In maintenance projects, however, the team begins by assuming success. That is, the software exists and is operational. Whether a software project overran its development budget or schedule, or didn't meet all the initially specified requirements is largely (but not totally) irrelevant. Success is now measured by the degree of change that the software can absorb. This measure is defined by answers to questions like these:

- How much customer value can we add in the form of functionality or performance enhancements?
- How often can we add this value?
- What is the operational quality of the enhancements made? Is there any backsliding on the current level of performance, reliability, look-and-feel, and so on?

The backsliding point is significant because users of existing software are classically averse to loss. That is, they feel the loss of existing software functionality or performance much more keenly than if they had never had it in the first place. How deeply a software system is embedded within a user's day-to-day work has major implications on risk and how they must be addressed.

Of course new software development may experience many of the same types of risks as maintenance, but not in the same combination. For example, an iterative development might find itself constrained by a poor design in a previous iteration, but there usually is time to cor-



rect it. Requirement changes also flood new developments, but if a requirement is deferred, there are usually no customers who can readily see their operations grinding to a halt because of the deferral. New hires may staff new developments, but they rarely form the preponderance of the development team. It is the cumulative effect of these risks that makes maintenance-related risk management different from, and usually more difficult than, development-related risk management.

Solutions. To address the root causes of maintenance risk—lack of control, information, and time—we changed the risk management process we used during the I-CASE PMS project to be significantly more customer-centric, real-time in nature, and sensitive to the impacts and interrelationships of individual risks. For instance, instead of waiting six to eight weeks to redo a risk assessment as in the I-CASE PMS project, we streamlined the assessment so that it could be performed weekly without undue stress to the release teams, or to the individual maintainers who had sole responsibility for part of a release.

Because maintenance risks have relatively higher immediate consequences than a similar development risk, we made the process more sensitive to individual risks, how risks coupled, and how they compounded. We also changed the identification process to be more perceptive to the risks and the acceptability of those risks, as NMSO's customers saw them.

We further strengthened our sensitivity to managing customer-related risks by developing a specialized risk management release process for use by NMSO's Change Control Board. Although the Board did assess the cost-benefit tradeoffs of requested enhancements and bug fixes, it was not quite as thorough as it could have been in considering the risks involved in a new release. NMSO management's promise to deliver more than they were able in the next release is one example. The intentions were good, but in trying to respond to every customer need, they were perpetuating the problem: With each late release, more was promised the next time, which made the next release even later, and so on.

Our goal was to break this pattern by instituting mechanisms that would help the Change Control Board better understand the level of risk inherent in a new release and the implications to the organization in terms of reputation, credibility, and so on. We also wanted the Board to understand what would happen when the risk became a problem, as described in the box on page 46, specifically how NMSO's customers could continue to operate the system as expected.

To this end, we had NMSO create, for each release, a risk profile that examines its objectives, assumptions, and constraints along with its political and technical feasibility. The Change Control Board can then use these results along with other information to determine when and what goes into a release. NMSO uses this release-specific risk management information as input to the risk management process.

IMPLEMENTATION SUPPORT

NMSO codified the risk management process we used in the I-CASE PMS project as well as the process modified for organization-wide use. The next step was to create an education and skills development program to help establish the processes. On the basis of what we found through the cultural assessment, we established the following goal for the program: Get the management and staff comfortable with the notion of risk, why it must be managed, and when, and then make sure they understand what benefits could accrue from the process.

We developed a series of lectures and seminars to educate and train the staff on the concepts and application of risk management. We developed a similar education program for NMSO's customers to help them understand how they influence the success of a proposed software change. At the project level, we created a skills development program aimed at providing just-in-time training to specific projects.

In support of these programs, NMSO's software engineering process improvement group teaches risk management courses and advises project teams and the Change Control Board.

MONITORING AND FEEDBACK

Although building the infrastructure and supporting it are critical, the entire effort would collapse without some mechanism for monitoring the progress of and obtaining feedback on the institutionalization process. NMSO periodically surveys its selected customers and personnel. During the survey, personnel are asked such questions as "How much training have you received?" "Are you using risk management in your daily work?" "Is it making a difference?" Customers are asked "Do you find NMSO's schedule promises more credible?" Is your confidence in NMSO increasing?" and so on.

New software development may experience many of the same types of risks as maintenance, but not in the same combination.

The responses highlight areas of concern. For example, if the survey reveals that adequate training has been received, risk management is being used, but the perception is that it makes no difference, NMSO digs deeper to identify why. With this information, we can modify the

processes, the policies for using them, and the level of training.

Customer responses are a critical part of this feedback. Only when the customers notice a difference will management support increase and institutionalization become something the organization wants to attain.

NMSO is in the very early stages of institutionalization. Several new projects and the Change Control Board are applying risk management. The staff is receiving both education and skills training. Benchmarking is proceeding, and the results are being used to determine what needs to be changed. We expect it will take another 18 to 24 months for NMSO to have a working organization-wide risk management policy.

Risk management for software maintenance is largely unexplored and we are still learning. In trying to establish a risk management policy across NMSO, we found that risk management for maintenance organizations, while similar, has significant differences from development-oriented risk management that must be taken into account. While these differences cannot be ignored, they could be significantly lessened if developers took the time to understand the risks they were creating for maintainers and tried harder to design software for less risky maintenance.

Finally we believe that the institutionalization process we describe here would be useful for introducing risk management and other improvement processes into organizations that focus on development. The cultural assessment is particularly important: Resistance to change is a universal obstacle. By overcoming it, the software community can get closer to the elusive goal of continuous process improvement. ◆

ACKNOWLEDGMENT

This essay represents the views of the authors alone and does not represent the views of any governmental organization.

REFERENCES

1. R. Charette, *Applications Strategies for Risk Analysis*, McGraw-Hill, New York, 1990.
2. S. Sherer, *Software Failure Risk*, Plenum Pub., New York, 1992.
3. C. Jones, *Assessment and Control of Software Risks*, Prentice Hall, Englewood Cliffs, N.J., 1994.
4. R. Charette, *Software Engineering Risk Analysis and Management*, McGraw-Hill, New York, 1989.
5. M. Carr et al., "Taxonomy-Based Risk Identification," Tech. Report CMU/SEI-93-TR-19, Software Eng. Inst., Pittsburgh, 1993.
6. F. Sisti and S. Joseph, "Software Risk Evaluation Method (ver. 1)," Tech. Report CMU/SEI-94-TR-19, Software Eng. Inst., Pittsburgh, 1994.
7. M. White, K. Adams, and F. Sisti, "Transforming Unseen Risks Into Focused Objectives," *Proc. 4th SEI Risk Management Conf.*, Software Eng. Inst., Pittsburgh, 1995.
8. E. Schein, *Organizational Culture and Leadership*, 2nd ed., Jossey-Bass, San Francisco, 1992.
9. D. Wilson, *A Strategy of Change*, Routledge, London, 1992.
10. A. Gemmer, "Risk Management: Moving Beyond Process," *Computer*, May 1997, pp. 33-43.
11. R. Charette, "The Risks with Risk Analysis," *Comm. ACM*, Vol. 34, No. 6, 1991, p. 106.
12. W. Humphrey, "Quality from Both the Developer and User Viewpoints," *IEEE Software*, Sept. 1989, pp. 84, 100.

Robert N. Charette is president of ITABHI Corp. in Fairfax, Virginia. His interests are in corporate entrepreneurialism, unified approaches to the proactive management of risk, systems engineering, and very large scale systems definition and management. He has consulted and lectured internationally and published numerous books on risk management.

Charette received a PhD in computer systems engineering from Kennedy-Western. He is the chairperson of the SEI risk management program advisory board and a member of the IEEE Computer Society and ACM.

Kevin MacG. Adams is a project engineer at QED Systems. Before joining QED, he was the systems engineering manager at the Navy Maintenance Support Office, where he evaluated new technologies and performed systems engineering and integration for the Naval Shipyard Corporation.

Adams received a BS in ceramic engineering from Rutgers University and an MS in naval architecture and marine engineering and an MS in materials engineering from the Massachusetts Institute of Technology. He is a member of the National Engineering Honor Society, the American Society of Naval Engineers, and the United States Naval Institute.

Mary B. White is a program manager for Andrulis Corp. Before joining Andrulis, she was a lead engineer for the Navy Maintenance Support Office's I-CASE PMS project. She has published and presented papers on software engineering, project management, I-CASE, and DoD logistics.

White received a BS in industrial and systems engineering with a certificate in computer science from the Georgia Institute of Technology. She is a member of the Institute of Industrial Engineers and the American Society of Naval Engineers.

Address questions about this article to Charette at PO Box 1929, Springfield, VA 22151; 75000.1726@compuserve.com.