Gray Rebuts Bach: No Cowboy Programmers!

Lewis Gray, Abelia Corporation

One of the hazards of sending me a rebuttal to one of my columns is that I may offer to publish it. This month, Lewis Gray has, like a true hero, stepped up to the challenge. My response follows.

—James Bach

James Bach's recent article "Microdynamics of Process Evolution" (*Computer*, Feb. 1998, pp. 111-113) moved me to write a response to the entire collection of methodological arguments it exemplifies. Along with other like-minded arguments, Bach's article promotes heroism as a substitute for process. My goal here is to point out an inherent biological limit to dependence on heroes.

Mountaineers who climb high mountains, like Mount Everest, experience an insidious debilitation, called *hypoxia*, that impairs judgment—even the ability to detect the impairment. Software managers and developers experience a common problem that is similar in some ways to hypoxia. It limits what heroes can be expected to do.

When managers take away process standards as development tools, they take away the very tools that developers need to cope with the problem.

When people argue that all process standards hinder software development, they're promoting a "cowboy" approach.

HYPOXIA AND STRESS

I have been reading a lot about Everest in the past year, including Jon Krakauer's *Into Thin Air* (Villard, 1997), a tragic story about the death of five people near the summit in 1996. Two of them were widely admired professional mountaineers and guides. All of them were fit and well trained. Their exceptional drive and focus pushed them through miserable conditions all the way to the top of the highest mountain on earth. Their behavior during the climb is what most of us mean by the word heroic.

Krakauer reports that a major factor in the deaths of these five heroic people was hypoxia, or lack of oxygen. High on Mount Everest, in the death zone above 25,000 feet, the amount of oxygen in the air drops to only one-third of what it is at sea level. When the human body is deprived of oxygen to this extent, it always breaks down. Even Sherpas in Nepal, for example, live well below the altitudes of the campsites on the way to the summit of Everest. Everyone who goes to the summit of Everest becomes seriously hypoxic.

Hypoxic people not only don't think well, they also don't know that they don't. Anticipating this problem in 1996, the guides set strict rules for how and when the group members should attempt to make their summit attempts. In effect, the rules were intended to replace judgment at the most dangerous part of the climb near the summit. One of the rules was to turn around and head back down the mountain at 2:00 p.m. on summit day, no matter how close to the summit anyone might be at that time.

In the everyday world, there is a common medical condition—called stress—that is similar in some ways to hypoxia. Heavy stress impairs our thinking and judgment. We find that we can't identify and weigh alternatives like we can when we're calm. As with hypoxia, under heavy stress, we often simplify our options into black-or-white problems with an obvious solution. Then we quickly seize the solution so we can get on to the next problem. It feels right, and we feel like efficient problem-solvers.

But from past experience, we all know that this approach to decision making is seductive and defective.

CHECKLISTS

There is a popular antidote for poor decision making under stress: Lists. All kinds of lists, from grocery lists to to-do lists, can help. Project managers use checklists to estimate and control projects. Pilots use checklists to prepare for flight. Scuba divers use checklists before going into the water.

Everyone uses lists for the same basic reason. We all recognize that when we're preoccupied, under pressure, or distracted we forget things and make errors in judgment. Lists are like the rules that the Everest climbers imposed before their climb. We need them to simulate good judgment at certain critical times.

Many modern software engineering standards, like ISO/IEC 12207 (Infor-

mation Technology—Software Life Cycle Processes), MIL-STD-498 (Software Development and Documentation), quality standards such as the ISO 9000 series, and process documents like the Capability Maturity Model (CMM), are just sample lists. Speaking as one of the designers of MIL-STD-498 and IEEE/EIA 12207, I can report that these standards were designed to be checklists of tasks to consider during software project planning.

PROCESS STANDARDS

In a development situation, you or I might choose not to do some task in a standard because it might not be appropriate for the project or organization. In many modern standards, the only truly mandatory activity is tailoring the standard to your particular needs.

Modern process standards are not designed to replace professional skill or experience in software development. Pilots know how to fly before they're hired by airlines, and they don't use checklists as do-it-yourself flying manuals.

No one should expect that standards like MIL-STD-498 or ISO/IEC 12207 are do-it-yourself software development manuals for novices. For the average software professional, process standards are "pilot checklists" for getting software development off the ground. The value of putting the tasks in a standard is that doing so forces standard users to acknowledge, and better yet to attempt to understand, the possible negative consequences of not doing the standard's tasks on their projects. This is the heart of the tailoring process and is a critical part of successful project planning.

Why use a standard when you can develop your own personal checklist? One reason is that hundreds or thousands of software professionals have contributed tens of thousands of comments designed to polish a standard like MIL-STD-498. It doesn't seem sensible to many people to ignore these insights completely and to start a list from scratch based only on personal, necessarily more-limited experiences.

COMPLIANCE WITH STANDARDS

So what about a hard-hearted auditor who objects to any deletion of any requirement in a standard such as the ISO 9000 series or the CMM? Perhaps the auditor won't let you tailor the standard even though you feel that some requirements are inappropriate to your particular project.

Doesn't an audit refute the claim that modern process standards are designed to be checklists for use as memory aids by

Continued on page 105

James Bach Responds: Thoughtful Practitioners Are Heroes

My article was about why we ought to pay careful attention to situational details and let situational problem-solving—rather than compliance to standards—drive software process evolution. In other words, I stressed the importance of the utility of a process. However, I did not devote much attention to process integrity: how well we adhere to our plan. (I did discuss process integrity in my first Software Realities column "The Hard Road from Methods to Practice," *Computer*, Feb. 1997, pp. 129-30.)

The *American Heritage* dictionary defines integrity as "the state of being unimpaired." The core of Gray's argument is really about process integrity. He is worried that we won't pay attention to best practices (as codified in standards) and that, under pressure, we won't even follow our own best judgment. In the death zone on Everest, or while preparing my golden master CD to ship to a client, lack of process integrity would be a real problem. In such situations utility and integrity are strongly linked. But I still say that utility reigns supreme.

Gray suggests that lists are an antidote to poor decision making under stress. But isn't that only true for lists that fit the situation at hand? "Turn around by 2:00 p.m." fits for highly trained mountaineers on Everest, but not for most climbers on most other mountains. For a climber like me, the rule would be "Don't climb Mount Everest." There are more generally applicable lists, but the more general they are, the less concrete they are, and that just throws us back to reliance on our own judgment.

Gray argues that standards can convey useful wisdom. He tells us that a list built from thousands of practitioners is better than one I build myself. I don't understand his logic. Surely if I had access to the experience, talent, and knowledge of thousands of practitioners, I'd be vastly more capable than I am. If only standards provided that! The process of creating a standard is a rich, lengthy dialogue among peers, full of creative disagreement, the actual thinking surrounding which almost never makes it into the standard. But the process of following a standard has nothing of that richness and thoughtfulness. It invariably becomes a process of complying with rules, no matter how "tailorable" a standard is supposed to be.

Gray fails to make the very important distinction between a list of things to think about and a list of rules. Lists of thought-ticklers can help us understand a problem better under pressure, while rules imply compliance and constrain behavior. There's a world of difference there. When we're under stress in a project, we need to behave more thoughtfully, not less so. We need to short-circuit rules and processes that don't apply, and protect those that do.

For Gray to lay the blame for poor use of standards on "unskilled standards users" is a bit extreme, since there is no defined skill set for standards users, no certification program for people who write standards, and no way of measuring the quality of a standard. By his logic, the fact that people have trouble quitting smoking can be attributed simply to unskilled smoking quitters.

I think he will agree with me on this: Just because something is called a standard doesn't make it useful, and no standard for an intellectual process can be useful unless applied by thoughtful practitioners. I call such people heroes.

benefits at the controls of enormous wealth-creating engines (this is world-class envy talking here).

How indeed? That's where Elvis enters the model.

ELVIS SIGHTED IN SILICON VALLEY

Perhaps the first thousand times he heard, "Elvis, you're the king!" the young singer blushed and smiled and felt flattered. After a few thousand times, he began to believe. Then he *became* the king and began to live the part. The loyal fans played their part. And Elvis came to believe he earned and deserved the position.

A king is more than a mere mortal. Believing he was more than mortal led Elvis to excesses a mortal body couldn't tolerate. Eventually, it killed him, and his premature death is partly our fault. The public conspires with entertainers to create this situation. We grow up on a steady diet of fairy tales and movies, so we're prone to believing them.

Imagine yourself as Elvis the executive at the controls of a high-tech corporation. Your business is seeing double-digit growth, everything you try works, you're moving up in the organization, your personal fortune is ballooning. The feedback is all positive.

You may begin to confuse market pull with personal leadership. And you're not talking to the actual creators of all this wealth (the nerds) because the gap is too great. Now that you're the king, you only talk to other executives, would-be executives, designated experts, and the press. What you don't know is that all these information sources are inherently flawed. Other executives aren't likely to have better information than you do. Would-be executives want to be promoted, which may cause problems with information quality. Designated experts are university professors and consultants who are outside the corporate class structure and are, therefore, eligible to interact with both the nerds and the executives. But the designated experts aren't the nerds—they're outsiders with information that lacks depth. And they may have their own agendas, so information quality is again lost.

Some version of Elvis was probably responsible for the decisions Intel made as it bungled the handling of the Pentium bug.

I don't have a fix for this problem. I'm not even sure anything needs to be fixed. It's the way things are; and the system seems to be working. I'm sitting back with the nerds enjoying the comical hot-topic cycle, occasionally contributing fuel, and waiting for my chance to become Elvis. ❖

Nick Tredennick has more than 22 years' experience in computer and microprocessor design. He enjoys nerd humor. Contact him at 1625 Sunset Ridge Rd., Los Gatos, CA 95030-9435; bozo@tredennick.com.

Software Realities

Continued from page 103

skilled software professionals? Doesn't an audit show that the standards are full of requirements that must be satisfied even when it doesn't make sense to do so, that they are really used to substitute the standard writer's judgment for the judgment of real people on the project?

In fact, it really doesn't. An audit is imposed (directly or indirectly) by buyers, who are customers. A company might voluntarily submit to an audit—an ISO 9000 audit, for example, to certify or register a quality system—but would only do so with the expectation that the audit results would favorably impress potential customers. Foolish buyers, or foolish auditors, might insist that developers do foolish things, and they might be more of a nuisance wielding a standard than they would be without it. But buyers and auditors are not under the control of the standard. Your organization can freely choose whether or not to enter into a contract with a buyer that makes foolish use of a standard.

Standards are not always used as intended. When this happens to a good standard, such as MIL-STD-498 or ISO/IEC 12207, let's place the blame where it belongs: on unskilled standards users.

There is a popular antidote for poor decision making under stress: Lists.

HYPOXIC HEROES ARE VULNERABLE

When people argue that all process standards hinder software development, as Bach does, they're promoting a "cowboy" approach that glorifies heroes. According to his logic, you can't be a hero using a process standard. There are no heroes without risk. It follows that the bigger the risk is, the bigger the hero, the more the stress.

Without process standards to nag them at times of stress, when they need them the most, cowboy developers will push past their biological limits with no help in sight. It's like putting climbers into the death zone on Everest with no rules for what to do on summit day.

Stress will cut away their competence. They won't notice. And because they rely only on themselves, they will make bad decisions. We've all been there. We've all done that. If I read Krakauer correctly, a big part of the reason that the climbers died on Everest in May 1996 was that, tragically, in their impaired, hypoxic state, they broke their own rules.

The lesson I see for software development is that organizations and projects need process standards the most when their people are most under pressure and have little time for thought. That is when everyone hits a biological limit and when it is most dangerous to let heroes run free without rules or guidelines. ❖

Lewis Gray is president of Abelia Corporation. He is also a software process improvement coach and long-time teacher of software development standards. Contact Gray at lewis@abelia.com