Software Process Improvement – A Good Idea for Other People

I heard a good joke the other day. The story goes that while in a Third World country, a guy grabbed a taxi to go to the airport. At the first red light, the taxi driver just whizzed through without even slowing down. When the passenger complained, the driver said, "Oh, my brother runs red lights all the time. He never has a problem!"

At a stop sign, the driver again whizzed through, never even bothering to look at the cross traffic. When the passenger complained again, the driver replied, "Oh, my brother never bothers to stop at stop signs. He never has a problem!"

Finally, the taxi approached a green light at an intersection. The driver slowed, them came to a full stop, and looked both ways. When the passenger asked why the driver was stopping now, the driver replied, "Well, you never know when my brother is coming! That makes it *our* problem."

The CROSSTALK theme this month is acquisition and supporting articles discuss software process improvement. Now, I'm the first to tell you that I don't really need a process, because I don't have a problem. After all, I'm Dave Cook! You don't know how great I am? Just ask me – or better yet, check out some examples of my work! Best coding you'll ever see. Other programmers blush in shame when they see the craftsmanship of my code. Design? Shoot, I can literally see the interfaces in my head. I'll make the code work, and get it to interface with your substandard code without any problems.

Requirements? Well, you just tell me what you want and I'll write it. You want to change the requirements? Just tell me. I'll fix it. I'm smart enough to understand the rules, and when necessary, to break them. If I violate the configuration management process occasionally, it's because it's in the best interest of the project.

Now, am I really that good? Well, those folks who have had the privilege of working with me will tell you I'm not[1]. And indeed, while I know that I am a good software engineer, I'm almost definitely not as good as I think I am. But the key is, I do think that I am!

Everybody is in favor of process improvement because other people need process discipline. I don't need it,

not me. I'm good. It's the others.

Not that I'm a prima donna – I just think that I'm a darn good developer. All developers think that about themselves. In fact, not a single developer gets up in the morning, looks at himself or herself in the mirror and says, "You know, I'm really not that good." Far from it. If asked, each developer would honestly think they are above average. Many think they are far above average.

Process improvement is something you do to protect yourself from others. If you don't require anybody to manage requirements, record and coordinate designs, etc., then nobody will.

How do you incorporate good software discipline and software processes? Well, I can sure give you some very good hints:

1. Start off with a meeting. Make sure it's two or three hours long. Grab the most boring, monotonous speaker you can find. Spend two or three hours droning on and on about how great the new process is – giving examples that have little or no relevance to what your particular developers do.
2. Tell your developers that they have to follow the process – or else! Give them no way to tailor the process to fit their needs. Don't dream of motivating them; just tell them what to do.
3. Implement the new process immediately throughout the company. For best results, require the use of a new software tool that no one really understands yet. Make sure that no one has more than minimal training on the tool, and that the experts on it are not on-site. In fact, the only available help should be restricted to a poorly designed Web page.
4. As mentioned earlier, make sure that if problems occur, change the work to make it fit the process. After all, the 20-plus years of experience your folks have are no match for a process or tool designed last month by somebody who once took a Java class in college. Don't consider that perhaps common sense is more important than blindly following the process.

Of course these are good hints. Not for you – for me! I'm a consultant, and I can use the work.

See you at the Systems and Software Technology Conference 2004.

— David A. Cook, Ph.D.
Senior Research Scientist
AEgis Technologies Group
dcook@aegistg.com

Can You BACKTALK?

Here is your chance to make your point, even if it is a bit tongue-in-cheek, without your boss censoring your writing. In addition to accepting articles that relate to software engineering for publication in CROSSTALK, we also accept articles for the BACKTALK column. BACKTALK articles should provide a concise, clever, humorous, and insightful article on the software engineering profession or industry or a portion of it. Your BACKTALK article should be entertaining and clever or original in concept, design, or delivery. The length should not exceed 750 words.

For a complete author's packet detailing how to submit your BACKTALK article, visit our Web site at <www.stsc.hill.af.mil>.

1. In fact, the associate publisher of CROSSTALK will probably go out of her way to tell you I'm not. Make sure you go to the Systems and Software Technology Conference 2004 and ask her!