



# Don't Just Kick the Tires

**Reuel Alder**  
*Publisher*



When I was the manager of a software development team, I worked for some difficult customers. Our processes were developing but in fairly good shape; however, our customers were difficult to service because they lacked discipline in developing or managing their requirements. They did not know what they needed or the priority of each requirement. Add to this that our software product served several user communities—all of whom similarly lacked a requirements management process—and you can imagine our distress. This forced us to assemble all user requests and assign our own priority. We would then work from the top down in hopes of completing the work before the dead-

line. This chaotic requirements “process” inevitably resulted in schedule and cost overruns.

I would like to say that this is an anomaly in defense software development, but it is not. Requirements generation for software-intensive systems is a widespread problem area. Requirements are at the beginning of the lifecycle product development and are therefore the most expensive to change or fix.

I wonder if this problem has a cultural basis. The American public has been trained by a sales-intensive environment to not analyze needs before making a major purchase. In a sense, we kick the tires, examine the pretty red paint, listen to a lot of evasive monthly payment jargon, then buy the car.

In making a sale, salespeople are trained to get customers emotionally

committed to a product before filling in the rational thinking. This technique works because few buyers enter the store knowing what they want. They depend on the salesperson to essentially tell them what they want. They want to be persuaded to make a purchase. If only they did their homework ahead of time, they could make a purchase more suited to their needs, not to mention their budget.

This cultural bias appears to feed the indecision and impulsive requirements development in today's defense acquisition community. With the need to conserve resources and stay battle ready, the Department of Defense needs the discipline of a requirements management process for buyers as well as for developers. Mature software development processes are highly effective, but they cannot compensate for a lack of requirements management by the user. ♦



## Letters to the Editor

### Experienced Developers as Mentors for Management?

Alan Reagan's comments (Letter to the Editor, September 1998) prompt a reply. First, for the past nine years, I have been a government contractor, involved in both the nuts and the bolts of software development and in software process improvement consulting. Second, I don't think of myself as “the enemy,” but believe that the roots of the problem lie elsewhere.

Software developers generally are engineers interested in building a working system that fills the customer's needs. Their goals (regardless of their CMM [Capability Maturity Model] or their ISO [International Organization for Standardization] status) are related to the so-called *ilities*: quality, reliability, usability, etc. Program managers, though, are often more concerned with budget and schedule issues: milestone achievement, acquisition reviews, etc. This does not imply that one set of priorities is wrong and the other is right or even that one has inherently more merit. It merely means that their priorities are different.

My point: While project managers may understand the factors and dynamics affecting their priorities, they often have little background in the technical side of software development. Rather than trying to make project managers into software engineers, we could rely on an experienced developer to act as guide (mentor?) through the swamp of software development.

And finally, if, as Reagan asserts, many of the independent validation and verification (IV&V) team members he has dealt with have fewer than five years experience, perhaps the request

for proposal for IV&V support should be rewritten to require proven development experience.

Joe Saur  
*SOCOM, Keane FedSysDiv*  
*Tampa, Fla.*

### Use It or Loose It

No one disputes the problems associated with software. Over the years, many solutions have been developed but applied sparsely. We now have the Software Engineering Institute, the Software Technology Support Center, and other organizations that provide outstanding packages. We also have organizations such as the Institute of Electrical and Electronics Engineers Computer Society, the Association for Computing Machinery, and the Institute for Certification of Computing Professionals that provide leadership in moving software engineering into a profession. Some of these efforts are directed toward the Department of Defense (DoD) and related industries and others toward process control or imbedded systems. I hope the developments will not be lost on the non-DoD business organizations, because software engineering is software engineering regardless of where it is applied. In fact, we may need these processes more here than in the areas for which they were developed. I also hope that the integration efforts, e.g., hardware, software, and communication, will continue and become the norm. We need total systems and not the finger pointing we seem to have now.

Ed Mechler  
*Project Controls, Inc.*  
*Penn Hills, Pa.*