Familiar Metric Management: Executive Backing

"Information development must be spearheaded by a general, not coordinated by aides-de-camp."

Peter G. W. Keen 1981 [fn]Peter G. W. Keen, "Information Systems and Organizational Change," *Communications of the ACM*, Jan. 1981, pp. 24-33.[/fn]

It was late in the afternoon of the final day of a conference on software about 20 years ago. The few remaining die-hards were listening to a panel of industry experts discussing software management. Finally the chair threw the session open to comments from the floor.

"The executives set us schedules that we can't meet," said one young man, adorned in beard and sandals, as was the custom of the period.

"They don't get the requirements right, nor give us time to straighten them out," said another.

"Frankly, they just don't understand software," said a third.

And so it went. We have been to a hundred conferences since then. In the corridors, birds-of-a-feather sessions, and late afternoon panels, we hear comments like these. The comments usually come from those deep in the ranks, probably without much first-hand knowledge of the pressures under which executives operate.

In the last year Scott Adams, cartoonist of Dilbert, has given public vent to these feelings. In a recent strip, for example, one of Adams software engineers tells his notoriously obtuse boss, the one with the pointy hair:

"You told me to finish my project in a week but its taken two months."

The boss listens to a series of such accusations for five more panels, culminating in the dig:

"Bottom line, your performance did not meet my expectations," the software engineer informs the pointy-haired one.

In the final panel the engineer returns to his cubicle, now all beat up, and tells Dilbert:

"It seemed like such a good idea."

Well, perhaps an approach this frank is not suited to the personalities of some bosses. What can we do?

It's a finite world

In a finite world it is an inescapable fact that those who allocate the resources, namely, executives, will labor under five pressures, diagrammed in Figure 1.

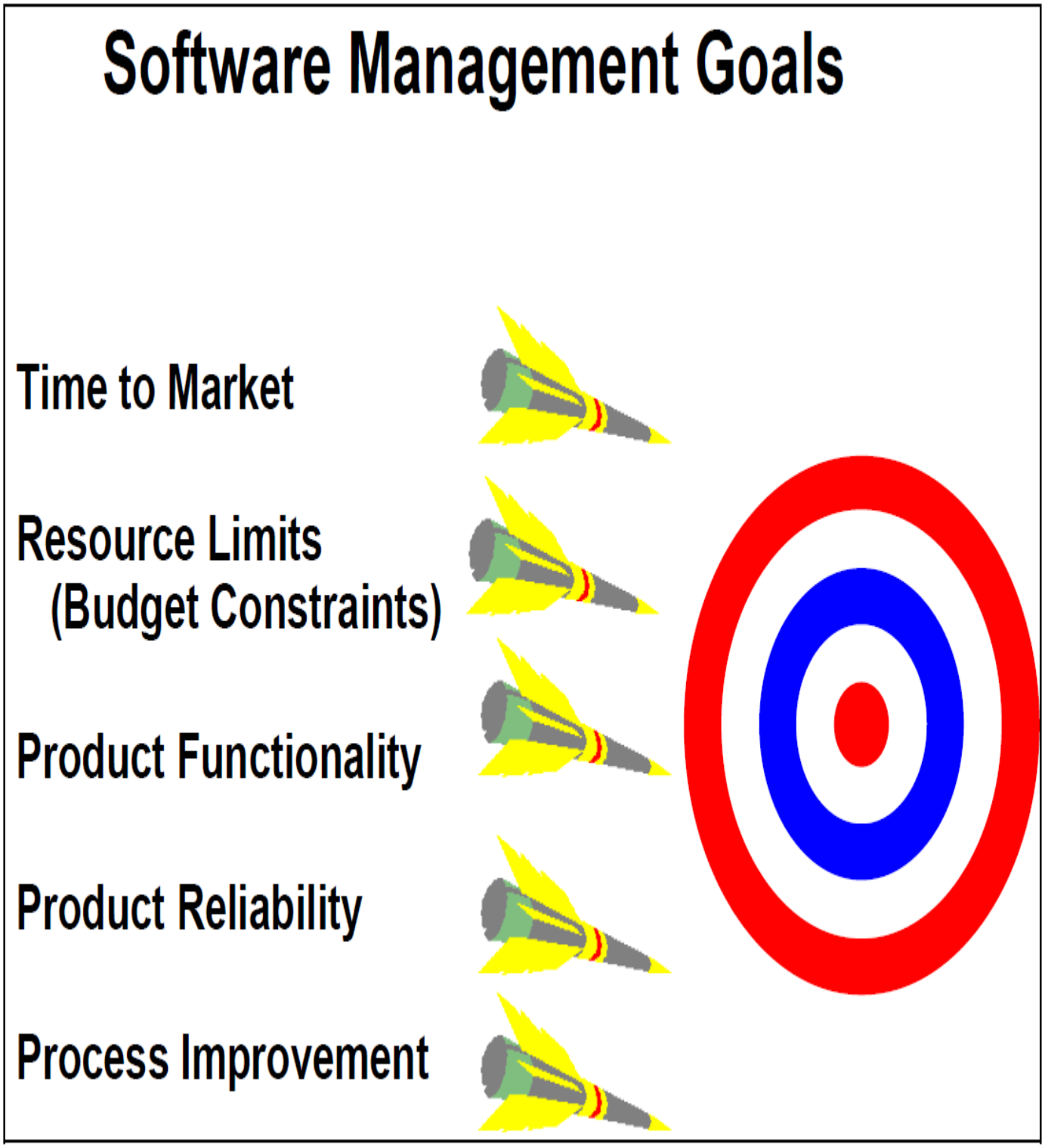

Figure 1. Executives try to meet five software management goals simultaneously.

The boys and girls in sandals (more likely now, in business dress) need to understand that these pressures exist. They are the inevitable accompaniments of a worldwide competitive society. Even in government-sponsored work, the taxpayers are setting limits on the funds available.

On their side, the resource allocators need to understand the gist of two arguments:

- The essence of software development
- The control of software development.

Remember, a level above the software director, the executive probably came up through marketing, hardware engineering, manufacturing, finance, or law. He lacks first-hand education, training, and experience in software development.

At this level, he or she may be managing five or six different functions, interfacing to a product market, responsible for raising capital funds, traveling worldwide, and trying to see his/her spouse and children occasionally. In short, he or she has little time for the niceties of software technology.

He or she may have a lot of experience, however, in trying to introduce new practices into organizations—total quality management, business process reengineering, the Software Engineering Institute's capability maturity levels, etc. They may even have heard of Howard Rubin's finding that 90 percent of the metrics programs initiated in data processing organizations are abandoned within the first two years. Or they may know something like this from their own exhausting experience. They know that making a new program work will be a labor of Hercules; they are not anxious to begin something they can't see through to a successful conclusion.

The Essence. Part of the responsibility lies with the software community to pull out of its complex technology just what the executive must know to function at his or her level, that is, to allocate resources effectively. How can we explain structured programming, object-oriented technology, client/server organization, the value of early inspections, the savings inherent in reuse, and so on, so that the executive can grasp what he needs to know in the little time he can spare? Admittedly, this task is difficult. Some of these technologies we are having trouble understanding ourselves. Yet, as software takes over the running of more and more of the world, we are greatly handicapped if those who allocate the resources do not understand what we do. That is essence. Equally important is how well we do it. That is control.

The control. Beyond essence lies control, and under control lies metrics. For example, development time measures time-to-market. Effort is a measure of the resources needed. Size reflects the product's functionality. Defect rate is a measure of product reliability which, in turn, is a key index of product quality. Process productivity measures the effectiveness of the software process.

These metrics give the executive (as well as the software engineers and their technology-conversant managers) the means to measure project execution. With these metrics, executives can raise their comfort level. They can see that estimates are realistic and that execution matches estimates.

Process productivity enables them to go one step further. With it they can measure their process and control, in the statistical control sense, its advancement.

Added value. Every function wants executive backing. In return, executives want to see "added value"—some kind of number demonstrating that the business is getting something added for the financial support they allocate to the function. Plaintive pleading for this support by software people is generally futile in the absence of good numbers. At the same time, executive actions motivated by emotion, such as firing the software director on personality evidence, do not by themselves assure that the software function will add more value the next time around the block.

Can we do anything about this standoff? Yes, many things. To name one, a new initiative of the IEEE

Computer Society.

Executive briefing

About two years ago the Society began to realize that many organizations had stalled on technical fronts for lack of executive understanding of the technologies involved. Publishing proceedings, compilations of papers, and technical books did not seem to be enough. The Society realized that it had a responsibility to reach the executive level with the essence of the technologies. This background understanding would then enable executives to generate the financial support and organizational impetus to advance the technologies with which the Society's members were concerned.

To get concrete, in October 1994, at the Computer Society's booth at OOPSLA 94 in Portland, Oregon, Matt Loeb, publisher of the Computer Society Press, outlined to Ware the kind of Executive Briefing he wanted from us. Ironically, as we talked, across the aisle several other publishers were selling books on object-oriented technology like hot cakes. It was evident that the technology-level people were doing their part to keep up. Matt wanted to reach the executive level, too.

An Executive Briefing, he said, was to be about 12 chapters of 2000 words each, with just one or two figures per chapter. It was to be keyed to executives not brought up in software technology. It was to be suitable for reading in the odd moments of busy executive life, such as on an airplane.

Now, in March 1996 that book, Executive Briefing: Controlling Software Development, has just been published by the Computer Society Press in Los Alamitos, California. The book does not deal with everything about software technology that an executive might find useful, the kind of knowledge that we have been calling essence here. What it does cover is the use of the five basic metrics to control development and enhance the process. It is a first step for the busy executive confronted with "the software crisis."