requirements

Editor: Suzanne Robertson ■ The Atlantic Systems Guild ■ suzanne@systemsguild.com

Risk Management during Requirements

Tom DeMarco and Tim Lister

We have heard a lot about the risk of not writing requirements, but little about how to profit from making risk management integral to the requirements process. Tom DeMarco and Tim Lister redress that imbalance by explaining the role that requirements must play in honest risk management.

—Suzanne Robertson

Risk management is project management for adults. This means the manager adopts an adult attitude toward things that might go wrong during the project, a marked difference from the prevailing can-do attitude. The risk manager is obliged to do some can't-do thinking, to look problems—even potentially unsolvable ones—directly in the eye and acknowledge that they could come to pass. The risk-aware project manager will accept a lucky break if it should happen but refuses to include it in the plan.

At the heart of risk management is a public, continuing process of risk identification. Some risks that will threaten your project are utterly unique to your situation. But others are not. Over some 10 years of conducting risk identification exercises in organizations, we have found five risks that are so ubiquitous that we've dubbed them *core risks*:

- *Intrinsic schedule flaw*: estimates that are wrong (undoable) from day one, often based on nothing more than wishful thinking
- *Specification breakdown*: failure to achieve stakeholder consensus on what to build
- *Scope creep*: additional requirements that inflate the initially accepted set
- *Personnel loss*: project members who leave before the project is done
- *Productivity variation*: difference between assumed and actual performance

All five of these touch on the requirements process, but the first two are deeply rooted there. The third sometimes indicates naturally occurring change, but in other cases is a direct indictment of how requirements were gathered in the first place. In other words, the so-called creep is not really change at all, but the correction of requirements that were misidentified to begin with.

We focus on the first two core risks and discuss risk management as part of the requirements process.

The schedule "requirement"

A schedule flaw can result from an honest mistake in sizing and projection. We estimate this to be true of approximately one project per million. In all the others, something else is going on. That something is most often a de facto elevation of a fervent desire ("Wouldn't it be nice if the project were done by year-end?") to requirement status. The elevation often goes one step further,

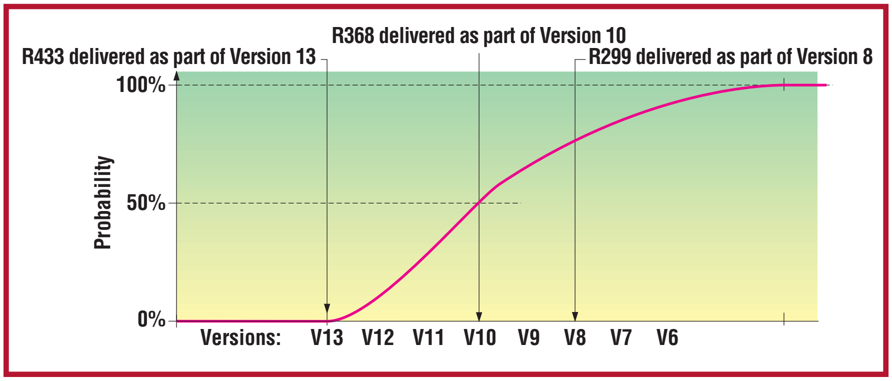

Figure 1. Risk diagram for a project with a fixed delivery date.

to nonnegotiable demand. When all or many of the other requirements are similarly nonnegotiable, the project becomes overconstrained. An overconstrained set of nonnegotiable demands—surprise, surprise—ends negotiation. When no trade-offs are possible, when it's not even permissible to discuss them, the project team is reduced to a single tactic: lying.

How can risk management help?

Risk management deals with this situation in two ways:

- It imposes a rank-ordering of all requirements (no two can have the same rank).
- It expresses all outcomes in probabilistic terms.

Now, if schedule is indeed the highest-priority requirement (R1), the chances of delivering rank-ordered requirements R2 thru R*n* would be presented in a graphic such as that in Figure 1. The figure suggests that completing V13 (containing R433 and all higher-ranked requirements) is only marginally probable, completing V10 (containing R368 and higher) is about 50 percent probable, and completing V8 (R299 and higher) is approximately 75 percent probable.

This *risk diagram* shows the probability of delivering any given version on or before the dictated end date. It's annotated to show which Rs are included in which versions. Completing an R implies satisfying all higher-priority (lower-number) Rs as well because rank-ordering requirements naturally leads to incremental implementation. Occasionally, a few lower-ranked requirements might be incorporated into a given version for convenience.

Vive la différence

When all parties agree to discuss success in probabilistic terms, they implicitly understand that no delivery probability in the real world ever equals 1.0. This forces all participants to confront risk. They might question how much risk there is, but never whether there is risk. Rank-order prioritization ensures that, however the project performs, the highest-priority requirements have the highest likelihood of being delivered.

Additionally, this approach helps flush out some of the pathologies that we all know happen when schedules are treated as absolute requirements.

- *Pathology 1*. The schedule is not really a requirement at all; it has been elevated to that status as a political statement. This is often clear from the fact that there is nothing particularly magical about the stated date (other than it being dear to the heart of some bigwig).
- *Pathology 2*. The schedule could be an overly precise statement of a requirement's result: "We need this really fast because we just heard our major competitor is going to have this in their next release and we think that 1 November just might be possible."
- *Pathology 3*. It could be an indirect statement of a constraint: "We want this, but we don't want to pay too much for it, so we figure that we can limit the team's spending if we give them only until 1 November."

When rank-order prioritization becomes the accepted way to establish each requirement's importance, there is little danger of overconstraining the project. Treating schedule as one of *n* priority-ranked requirements lessens the possibility of intrinsic schedule flaw. It also sidesteps the tendency to use schedule as a de facto cost reduction tactic. This approach alone won't make schedule flaws disappear, but it should help reduce their incidence to those that are honest sizing and projection errors.

Failure to concur

In the past, IT projects were most often tasked to satisfy a single user's requirements. They were relatively easy, but sadly we finished all such projects years ago. Today a new IT project is likely to affect several different stakeholders from different parts of the organization, in different locations, with different interests, and little or no common stake. Perhaps the biggest core risk is that these stakeholders will fail to concur on project goals. Our data leads us to expect this to happen to a disruptive extent on approximately one project out of seven.

Failure to achieve total concurrence would be no more than an annoyance if all could agree to disagree. We would end up delivering products that satisfied different stakeholders to differing degrees, with no one left completely out. Unfortunately, nonconcurring projects seldom play out this way.

The problem is that organizational culture might require all the stakeholders to cooperate, or at least *seem* to cooperate. This does not make dissent go away, but forces it underground. And dissent always exists—count on it. New IT products introduce change into organizations, and change is never uniform in its impact on different constituencies. Our basic rule is

> *Every time an IT product is delivered, somebody gains power and somebody else loses power.*

Both the power gainers and the power losers are, by definition, stakeholders.

You can expect some stakeholders on any complex project to be adversaries. They might not be allowed to act adversarially, but many other possibilities are open to them. For example:

- *Overloading*: adding excessive functionality to burden the project
- *Power broking*: forcing priority based on clout rather than need
- *Withholding*: refusing to sign off on functionality that benefits others
- *Unavailability*: being too busy to attend to project needs

Disguising nonconcurrence

A common tactic requirements gatherers use is to avoid confronting stakeholder nonconcurrence: They write down the requirements so vaguely that all the stakeholders can sign off on them. We usually categorize this as a simple writing problem (Gee, I wish those guys could write better!), but it isn't. Producing a specification that is vague enough to get everyone to sign but still looks like a specification requires huge talent. People who can pull this off are geniuses. It's too bad that their genius doesn't help the project. It just defers the real problem until implementation time, when it becomes fatal. You can specify a product vaguely, but you can't implement it vaguely.

Managing the risk of nonconcurrence

Failure to concur is clearly a political matter, not a technical matter. The technicians who make up much of our requirements engineering force are not naturally endowed with the skills and talents to deal with this.

Risk management cannot assure a solution to nonconcurrence either, but it can help detect it. The simplest way is to look for signoff on the boundary characteristic of the project's product—a complete census of inputs and outputs across the system boundary, defined down to the data element level. When this census becomes integral to the requirement, the parties cannot sign off without truly agreeing.

Because the boundary characteristic foils the most common defense against nonconcurrence (using a vague statement of requirements to gain signoff), you can expect all participants in a nonconcurring project to rail against its inclusion as a milestone. Expect them to say that it is a matter that should be left until detailed design or to the programmers. The risk manager must stand firm on this matter. Projects that can't achieve a signoff on the boundary census by the approximate 15 percent point probably need to be cancelled.

Risk management doesn't make any of the requirements-related risks go away. What it does is let us deal with these matters in a clear-headed, adult fashion.

Tom DeMarco and **Tim Lister** are Principals of the Atlantic Systems Guild and coauthors of *Waltzing with Bears: Managing Risk on Software Projects* (Dorset House, 2003) and *Peopleware: Productive Projects and Teams*, (Dorset House, 1987 and 1999). Contact them at tdemarco@systemsguild.com and tlister@systemsguild.com

PURPOSE The IEEE Computer Society is the world's largest association of computing professionals, and is the leading provider of technical information in the field.

MEMBERSHIP Members receive the monthly magazine **COMPUTER**, discounts, and opportunities to serve (all activities are led by volunteer members). Membership is open to all IEEE members, affiliate society members, and others interested in the computer field.

COMPUTER SOCIETY WEB SITE
The IEEE Computer Society's Web site, at **http://computer.org**, offers information and samples from the society's publications and conferences, as well as a broad range of information about technical committees, standards, student activities, and more.

BOARD OF GOVERNORS
Term Expiring 2003: *Fiorenza C. Albert-Howard, Manfred Broy, Alan Clements, Richard A. Kemmerer, Susan A. Mengel, James W. Moore, Christina M. Schober*
Term Expiring 2004: *Jean M. Bacon, Ricardo Baeza-Yates, Deborah M. Cooper, George V. Cybenko, Haruhisha Ichikawa, Lowell G. Johnson, Thomas W. Williams*
Term Expiring 2005: *Oscar N. Garcia, Mark A Grant, Michel Israel, Stephen B. Seidman, Kathleen M. Swigger, Makoto Takizawa, Michael R. Williams*
Next Board Meeting: *22 Nov. 2003, Tampa, FL*

IEEE OFFICERS
President: MICHAEL S. ADLER
President-Elect: ARTHUR W. WINSTON
Past President: RAYMOND D. FINDLAY
Executive Director: DANIEL J. SENESE
Secretary: LEVENT ONURAL
Treasurer: PEDRO A. RAY
VP, Educational Activities: JAMES M. TIEN
*VP, Publications Activities:*MICHAEL R. LIGHTNER
VP, Regional Activities: W. CLEON ANDERSON
VP, Standards Association: GERALD H. PETERSON
VP, Technical Activities: RALPH W. WYNDRUM JR.
IEEE Division VIII Director JAMES D. ISAAK
President, IEEE-USA: JAMES V. LEONARD

IEEE COMPUTER SOCIETY

COMPUTER SOCIETY OFFICES
Headquarters Office
1730 Massachusetts Ave. NW
Washington, DC 20036-1992
Phone: +1 202 371 0101 • Fax: +1 202 728 9614
E-mail: hq.ofc@computer.org
Publications Office
10662 Los Vaqueros Cir., PO Box 3014
Los Alamitos, CA 90720-1314
Phone:+1 714 821 8380
E-mail: help@computer.org
Membership and Publication Orders:
Phone: +1 800 272 6657 Fax: +1 714 821 4641
E-mail: help@computer.org
Asia/Pacific Office
Watanabe Building
1-4-2 Minami-Aoyama,Minato-ku,
Tokyo 107-0062, Japan
Phone: +81 3 3408 3118 • Fax: +81 3 3408 3553
E-mail: tokyo.ofc@computer.org

IEEE

EXECUTIVE COMMITTEE
President:
STEPHEN L. DIAMOND*
Picosoft, Inc.
P.O.Box 5032
San Mateo, CA 94402
Phone: +1 650 570 6060
Fax: +1 650 345 1254
s.diamond@computer.org

President-Elect: CARL K. CHANG*
Past President: WILLIS. K. KING*
VP, Educational Activities: DEBORAH K. SCHERRER (1ST VP)*
VP, Conferences and Tutorials: CHRISTINA SCHOBER*
VP, Chapters Activities: MURALI VARANASI†
VP, Publications: RANGACHAR KASTURI †
VP, Standards Activities: JAMES W. MOORE†
VP, Technical Activities: YERVANT ZORIAN†
Secretary: OSCAR N. GARCIA*
*Treasurer:*WOLFGANG K. GILOI* (2ND VP)
2002–2003 IEEE Division VIII Director: JAMES D. ISAAK†
2003–2004 IEEE Division V Director: GUYLAINE M. POLLOCK†
2003 IEEE Division V Director-Elect: GENE H. HOFFNAGLE
Computer Editor in Chief: DORIS L. CARVER†
Executive Director: DAVID W. HENNAGE†

* voting member of the Board of Governors
† nonvoting member of the Board of Governors

EXECUTIVE STAFF
Executive Director: DAVID W. HENNAGE
Assoc. Executive Director:
ANNE MARIE KELLY
Publisher: ANGELA BURGESS
Assistant Publisher: DICK PRICE
Director, Administration: VIOLET S. DOAN
Director, Information Technology & Services:
ROBERT CARE
Manager, Research & Planning: JOHN C. KEATON