from the best-selling authors of PEOPLEWARE

Waltzing *with* Bears

MANAGING RISK ON SOFTWARE PROJECTS

TOM DEMARCO & TIMOTHY LISTER

Waltzing *with* Bears

Managing Risk on Software Projects

Also Available from Dorset House Publishing

The Deadline: A Novel About Project Management
by Tom DeMarco
ISBN: 978-0-932633-39-2 Copyright ©1997 320 pages, softcover

Dr. Peeling's Principles of Management:
Practical Advice for the Front-Line Manager
by Nic Peeling
ISBN: 978-0-932633-54-5 Copyright ©2003 288 pages, softcover

More Secrets of Consulting: The Consultant's Tool Kit
by Gerald M. Weinberg
ISBN: 978-0-932633-52-1 Copyright ©2002 216 pages, softcover

Peopleware: Productive Projects and Teams, 2nd ed.
by Tom DeMarco and Timothy Lister
ISBN: 978-0-932633-43-9 Copyright ©1999 264 pages, softcover

Project Retrospectives: A Handbook for Team Reviews
by Norman L. Kerth foreword by Gerald M. Weinberg
ISBN: 978-0-932633-44-6 Copyright ©2001 288 pages, softcover

The Psychology of Computer Programming: Silver Anniversary Edition
by Gerald M. Weinberg
ISBN: 978-0-932633-42-2 Copyright ©1998 360 pages, softcover

Quality Software Management, Vol. 4: Anticipating Change
by Gerald M. Weinberg
ISBN: 978-0-932633-32-3 Copyright ©1997 504 pages, hardcover

Why Does Software Cost So Much?
(And Other Puzzles of the Information Age)
by Tom DeMarco
ISBN: 978-0-932633-34-7 Copyright ©1995 248 pages, softcover

For More Information

✔ Contact us for prices, shipping options, availability, and more.

✔ Visit Dorsethouse.com for excerpts, reviews, downloads, and more.

DORSET HOUSE PUBLISHING
An Independent Publisher of Books on
Systems and Software Development and Management. Since 1984.
3143 Broadway, Suite 2B New York, NY 10027 USA
1-800-DH-BOOKS 1-800-342-6657
212-620-4053 fax: 212-727-1044
info@dorsethouse.com www.dorsethouse.com

Waltzing *with* Bears

Managing Risk on Software Projects

Tom DeMarco & Timothy Lister

Dorset House Publishing
3143 Broadway, Suite 2B
New York, NY 10027

Library of Congress Cataloging-in-Publication Data

DeMarco, Tom.
Waltzing with bears : managing risk on software projects / Tom DeMarco & Timothy Lister.
p. cm.
Includes bibliographical references and index.
ISBN 0-932633-60-9 (soft cover)
1. Computer software--Development--Management. 2. Risk management. I. Lister, Timothy R. II. Title.
QA76.76.D47D4755 2003
005.1'068--dc21

2003043481

Cover Illustration: Tom DeMarco, detail from "Beach Dreams"
Author Photograph: James Robertson
Cover Design: David W. McClintock

Distributed in the English language in Singapore, the Philippines, and Southeast Asia by Alkem Company (S) Pte. Ltd., Singapore; in the English language in India, Bangladesh, Sri Lanka, Nepal, and Mauritius by Prism Books Pvt., Ltd., Bangalore, India; and in the English language in Japan by Toppan Co., Ltd., Tokyo, Japan.

Printed in the United States of America

Library of Congress Catalog Number: 2003043481

ISBN: 978-0-932633-60-6 12 11 10

ACKNOWLEDGMENTS

Many people believe that the roles of editor and publisher consist of checking spelling and grammar and overseeing the printing process. No. We thank David McClintock, Wendy Eakin, Vincent Au, and Nuno Andrade of Dorset House for their work taking our manuscript and molding it, transforming it, and coaxing it into a book we two are very proud of. The "before and after pictures" are remarkable. Thank you.

We also want to thank our colleagues who have generously jumped at the chance to give us their opinions and insights pro bono. It is just these conversations that give us the joy of working in our profession. Our thanks go to Rob Austin, Barry Boehm, Christine Davis, Mike Evans, Sean Jackson, Steve McMenamin, and Mike Silves.

We especially thank Bob Charette and the late Paul Rook for trailblazing this area. Our path has been much easier to follow thanks to them.

Finally, we thank our consulting clients of the last ten years. These are the companies that demonstrated to us that running away from risk is a loser, and that risk comes with the territory of a valuable project. We recognize that these people are not afraid to work on risky efforts; they want to work on important ventures.

To Sally O. Smyth and Wendy Lister
Risk Managers Extraordinaire

AUTHORS' NOTE

This text is divided into five parts, each one intended to answer one of the major questions likely to be on the mind of a new or potential risk manager:

Part I:	Why bother to do risk management?
Part II:	Why shouldn't we do it? (Wherein the authors come clean about some of the potential negatives of introducing risk management into an organization that isn't quite ready for it.)
Part III:	How shall we go about it?
Part IV:	How much risk should our organization be willing to take?
Part V:	How do we know whether or not our risk management approach is working?

The page introducing each new part breaks down the overall question into detailed questions. By reading the chapters in each part, you should find answers to all those questions—or we haven't done our job.

Voice

Most of the text is written in the plural voice, with "we" standing for both authors. On occasion, we like to get in a word or two in

our individual voices, and that gives rise to paragraphs set off like these:

TRL: Here's me (Tim) speaking in my own voice.

TDM: And this one is me (Tom).

Website

As we mention later, in Chapter 12, we've built a Website to complement the text. You'll find it at

http://www.systemsguild.com/riskology

We have placed some tools there to help your risk management effort, and we will endeavor to keep the site updated as we learn about new risk management tools or news on the subject.

Our Title

Our title is taken from a song included in *The Cat in the Hat Songbook,* by Dr. Seuss.[1] The song tells of Uncle Terwilliger, who every Saturday night "creeps down our back stairs,/sneaks out of our house to go waltzing with bears."

Uncle T. is a willing risk taker—we can only hope that he has a workable understanding of risk assessment, containment, and mitigation. If so, he is a perfect model for managers of risky software projects, people who may need to dance on occasion with a few bears of their own.

---

[1]Dr. Seuss and Eugene Poddany, *The Cat in the Hat Songbook* (New York: Random House, 1967).

CONTENTS

Prologue: The Ethics of Belief 3

PART I: WHY 7

1. Running Toward Risk 9
2. Risk Management Is Project Management for Adults 15
3. Denver International Airport Reconsidered 22
4. The Case for Risk Management 29

PART II: WHY NOT 35

5. The Case Against Risk Management 37
6. The Onus of Uncertainty 42
7. Luck 46

PART III: HOW 51

8. Quantifying Uncertainty 53
9. Mechanics of Risk Management 60
10. Risk Management Prescription 73
11. Back to Basics 80
12. Tools and Procedures 91

13. Core Risks of Software Projects 101
14. A Defined Process for Risk Discovery 113
15. Risk Management Dynamics 121
16. Incrementalism for Risk Mitigation 128
17. The Ultimate Risk Mitigation Strategy 138

PART IV: HOW MUCH 143

18. Value Quantification 147
19. Value Is Uncertain, Too 151
20. Sensitivity Analysis 156
21. Value Offsets Risk 160
22. Refining the Risk Management Prescription 164

PART V: WHETHER OR NOT 169

23. Test for Risk Management 171

Appendix A: The Ethics of Belief, Part 1 175

Appendix B: Risk Template 181

References 183

Index 191

Waltzing *with* Bears

Managing Risk on Software Projects

This page intentionally left blank

PROLOGUE

THE ETHICS OF BELIEF

London, April 11, 1876: The scene is Grosvenor Square, just before 10 P.M. Around us, on the sidewalks of the square, Victorian gentlemen, many in top hats and evening clothes, are making their way toward the ornate entrance of the Grosvenor Hotel. We follow them in and are guided toward the upstairs parlor, where the monthly meeting of London's elite Metaphysical Society is to take place.

The Society's members include Alfred Tennyson, William Gladstone, Thomas Huxley, Cardinal Manning, Arthur James Balfour . . . in short, the cream of London intelligentsia. The subject this evening is, as always, philosophy. Before the proceedings begin, the participants are talking in small groups, picking up threads of the last meeting's discussion. As we wander among these clusters, we hear such terms as ontology, tautology, and epistemology. Some of the discussions are heated.

There is a certain tension in the room this evening, due to the selection of the meeting's featured speaker. He is the Society's newest member, William Kingdon Clifford. Clifford is a professor of logic and mathematics at London's University College. He is considered an iconoclast, possibly an atheist, and is known to be a fiery debater. With his selection, he has become the youngest person ever accepted into the Society.

By convention, each new member must prepare a paper and read it to the membership at his first meeting. Only the title of

Clifford's paper, "The Ethics of Belief," has been made public, not the paper's contents. It promises to be a stunner.

Indeed, before Clifford has even finished reading, half the room has stomped out in angry protest. The Society's Secretary has publicly resigned; it would have been his job to arrange a private printing of the paper, and this he has refused to do. The remaining members are on their feet, either cheering Clifford on or trying to shout him down. The temperature in the room has shot up markedly and the entire scene is, well, a bit *un-British.*

What was it about "The Ethics of Belief" that got the members so hot? In the essay, Clifford asserts that what you choose to believe ought not to be exempt from the ethical judgment of others. Your belief may open you to a charge of unethical behavior, depending on whether, in Clifford's words, you have "a right to believe" the thing that you believe.[1]

He offers as an example the owner of an emigrant ship that is about to set sail with a full complement of passengers. The owner is bothered by worries that the ship is old and in poor condition and wasn't built very well in the first place. There is a real question in his mind about whether it can safely make another passage. With a bit of effort, though, the shipowner overcomes his doubts and persuades himself that no great harm will come from just one more passage. The ship, after all, has weathered more than a few storms in its day and always managed to limp home to port. Why not one more time?

The ship puts to sea and is lost with all hands.

"What shall we say of the owner?" Clifford asks, and gives his own answer:

> *Surely this, that he was verily guilty of the death of those men. It is admitted that he did sincerely believe in the soundness of his ship; but the sincerity of his conviction can in no wise help him, because he had no right to believe on such evidence as was before him. He had acquired his belief not by honestly earning it in patient investigation, but by stifling his doubts. And although in the end he may have felt so sure about it that he could not think otherwise, yet inasmuch as he had knowingly and willingly worked himself into that frame of mind, he must be held responsible for it.*

---

[1]See Appendix A for Part 1 of "The Ethics of Belief."

Clifford then goes back over the same story and alters it slightly. Suppose, he tells us, that the ship had managed after all to complete the voyage with no loss of life. Would the owner have been less guilty?

> *Not one jot. When an action is once done, it is right or wrong forever; no accidental failure of its good or evil fruits can possibly alter that. The man would not have been innocent, he would only have been not found out. The question of right or wrong has to do with the origin of his belief, not the matter of it; not what it was, but how he got it; not whether it turned out to be true or false, but whether he had a right to believe on such evidence as was before him.*

Prior to Clifford, there was a presumption that your beliefs could never be considered in an ethical light. You could believe any damn thing you pleased. You could even believe impossible things, as the White Queen did in *Through the Looking Glass*. When Alice protests that one simply cannot believe impossible things, the Queen responds,

> *"I daresay you haven't had much practice. . . . When I was your age, I always did it for half-an-hour a day. Why, sometimes I've believed as many as six impossible things before breakfast."*

There is probably no job on earth for which an ability to believe six impossible things before breakfast is more of a requirement than software project management. We are routinely expected to work ourselves into a state of believing in a deadline, a budget, or a performance factor that time subsequently may prove to be impossible.

We do this in a process that's not so terribly different from when the shipowner talked himself into believing in his ship. You have almost certainly been through this process yourself one or more times. There may have been others, egging you on. Your boss, for example, asks you to consider taking on a project that has to be done by Christmas, with only three people available to work on it. You express doubts that there is enough time to get the software built.

"That's why I picked *you* to manage the job," your boss tells you, confidently.

The fix is in: You'll get the job, the challenge, and the prestige . . . but you'll have to believe in the schedule. That's the price you pay. You swallow hard and say you'll do it. Later, you bolster your belief. Sure, why not Christmas? Other projects have accomplished as much in as little time, haven't they? Before long, you may find yourself actually feeling confident. Time may prove otherwise, but for the moment, you are practically sure you can get the job done.

At that moment, though, William Kingdon Clifford's question should be coming back to haunt you. Yes, that's what you believed, *but did you have any right to believe it?* Did you have a right to believe in that schedule, based on the evidence that was before you?

The business of believing only what you have a right to believe is called *risk management*. This essential discipline applies Clifford's ethics of belief to any effort that is complicated by elements of uncertainty. It will guide you through that effort (a software project, for example) in a way that eliminates the fabric of little lies and self-deceptions that have so hampered your work in the past. It will become your alternative to believing "six impossible things before breakfast."

3

DENVER INTERNATIONAL AIRPORT RECONSIDERED

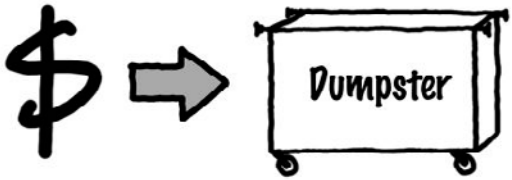
The city of Denver, Colorado, set out in 1988 to build a new airport to replace the existing one, Stapleton Airport. Stapleton was judged incapable of expansion, inadequate to serve the growing city, and guilty of contributing to ever-more-evident noise- and air-pollution problems. With the new airport, costs would be reduced, pollution and air-traffic delays would be eliminated, and growth would be assured. The new Denver International Airport (DIA) was scheduled to open on October 31, 1993. That was the plan.

Another Fine Mess

Cut to the chase: Everything went fine, except those damn software guys let the side down again. (Sigh, groan, general rolling of the eyes.) On October 31, 1993, every other part of the vast airport complex was ready to go . . . honest it was. Really. Trust us on this. But the software wasn't ready, so the airport couldn't open!

Specifically, what wasn't ready on time was the infamous DIA Automated Baggage Handling System (ABHS). The airport couldn't open without functional baggage-handling software. Since building the airport involved huge capital expenditure, all that capital was tied up while the software guys scrambled around playing catch up. And

time is money. The taxpayers took the hit. This is not a matter subject to elaborate analysis; it is as simple as this:



And it was all the fault of those awful software people.

This kind of dollars-to-dumpster simplification was a feature of newspaper and journal coverage of the DIA troubles from the first sign of delay in early 1993 until the partial opening in 1995. So much blame was laid on the software team that even today, the phrase "DIA Automated Baggage Handling System" is a recognized symbol of incompetent software projects.

An article in *Scientific American* put responsibility for the DIA disappointment squarely on the software industry and its lax standards and practices:

> *software engineering discipline remains years—perhaps decades—short of the mature engineering discipline needed to meet the demands of an information-age society.*[1]

This was a process problem, the article asserted. The delays at DIA might very well have been avoided, the article claimed, if only the project had improved its process to include

1. higher CMM level
2. more use of formal methods
3. mathematical specification languages like B and VDM

But was it really a process problem?

Beyond the Process

Suppose you had an utterly perfect process for delivering software. Would that remove all uncertainty from your projects? In

[1]W. Wayt Gibbs, "Software's Chronic Crisis," *Scientific American* (September 1994), p. 84.

fact, is the software building process even one of the *major* sources of uncertainty? We suggest not. Among the more important sources of uncertainty are these:

1. *Requirement:* What exactly is it that the system has to do?
2. *Match:* How will the system interact with its human operators and other peer systems?
3. *Changing environment:* How will needs and goals change during the period of development?
4. *Resources:* What key human skills will be available (when needed) as the project proceeds?
5. *Management:* Will management have sufficient talent to set up productive teams, maintain morale, keep turnover low, and coordinate complex sets of interrelated tasks?
6. *Supply chain:* Will other parties to the development perform as hoped?
7. *Politics:* What is the effect of using political power to trump reality and impose constraints that are inconsistent with end-project success?
8. *Conflict:* How do members of a diverse stakeholder community resolve their mutually incompatible goals?
9. *Innovation:* How will technologies and approaches unique to this project affect the eventual outcome?
10. *Scale:* How will upscaling volume and scope beyond past experience impact project performance?

Even the most perfect construction process can't remove uncertainty from a complex systems development project. Where there is uncertainty, there is risk. Where there is risk, there needs to be a conscious and thoughtful effort to manage it. Instead of asking, "How did they go about building their software?" we can gain a lot more insight into what happened at DIA by asking, "How did they go about managing their risks?"

Risk Management at DIA

In our brief summary of the events at DIA, we asked you to swallow the often-repeated claim that the airport was 100-percent ready to open except for the baggage-handling software, and that

the airport couldn't open at all without that software. Let's go over that premise again in some detail.

First of all, maybe the assertion that all the other subprojects were complete wasn't true. Maybe the baggage system was not the only late component, merely the *most visibly* late component. Maybe the whole schedule was hopeless and *everybody* was late. When this happens, a common ploy is for heads of the various subprojects to play a little brinkmanship to assert complete readiness, hoping that one of their peers will crack first. When someone finally cracks, the others just affect to wrinkle their brows in disappointment and then frantically use the extra time to fix up their own domains. Maybe that's what happened at DIA. But just for the purposes of this analysis, let's assume not. Take all the other subproject managers at their word and assume that the airport could indeed have opened but for the failure of the Automated Baggage Handling software. The entire cost of delay—more than $500 million in extra financing—was therefore attributable to the lateness of that one key element.

And now start asking yourself a few key questions:

Q1: Why couldn't the airport open without the baggage-handling software?
That's easy: The baggage-handling software was on the overall project's critical path for the airport's opening. It was so essential to airport operations that the members of the organization's governing board knew they couldn't move passengers through the airport, even for a single day, without that system.

Q2: Why was the ABHS on the critical path?
Well, because there was no other way to move the baggage. The system of tele-carts and bar-code readers and scanning devices and switch points and cart unloaders was the only way to get baggage to and from the planes.

Q3: Are there no alternative ways to move baggage?
Of course. There is, for example, the time-honored method of having big burly guys haul the stuff. There is also the conventional airport approach of small trucks pulling hand-loaded carts, daisy-chained together.

Q4: When the ABHS wasn't ready on time, why couldn't DIA open with one of these alternative methods of moving baggage?
Um. Well. (Hem and haw.) The tunnels that were meant to serve the automated tele-cart system were too low for people and couldn't accommodate the trucks. So the automated system had to work.

Q5: Couldn't the tunnels have been redesigned so that trucks and hauled carts could go through them?
Yes, but there wasn't time. By the time it was discovered that the ABHS software would be late, the tunnels were already built. And the time to revamp them was judged to be longer than the time required to perfect the software.

Q6. Couldn't the revamping of the tunnels have started earlier?
Yes, but that wasn't judged appropriate. Money and time spent on the tunnels would have been wasted had the software actually been delivered on time, as upper management was then assuring it would be.

Q7: Wasn't lateness of the ABHS software seen as a potential risk?
Only after it happened. Before that, the software was placed on an aggressive schedule and managed for success.

Q8: Haven't software projects been late before?
Yes, but this one was supposed to be different.

Q9: Was there any history of prior projects building similar systems?
Yes. The Franz Josef Strauss Airport in Munich had installed a pilot ABHS, designed along the lines of the DIA version.

Q10: Did the DIA team visit the Munich project, and if so, what did it learn?
Members of DIA's ABHS project did visit Munich. The Munich software team had allowed a full two

years for testing and six months of 24-hour operation to tune the system before cut-over. They told the DIA folk to allow that much or more.

Q11: Did DIA management follow this advice?
Since there wasn't time for such extensive testing and tuning, they elected not to.

Q12: Did the project team give sufficient warning of impending lateness?
First of all, the invisible hand of the marketplace made a significant gesture right at the outset. When the DIA board of governors first put the ABHS out to bid, nobody was willing to submit a bid for the scheduled delivery date. All bidders judged that starting the project off with such a schedule was a sure way to court eventual disaster.

Eventually, the airport engaged BAE Automated Systems to take on the project on a best-efforts basis. During the project, the contractor asserted early and often that the delivery date was in jeopardy and that the project was slipping further behind with each month and each newly introduced change. All parties were made aware that they were trying to do a four-year project in two years, and that such efforts don't usually come home on time. All of this evidence was ignored.

Risk Management Practices Honored in the Breach

It's not *how* risk management was practiced at DIA that sunk the project. It's that there was no effort at risk management at all. Even the most perfunctory risk management effort—probably in the first minute of the first risk-discovery brainstorm—would have listed a delay in the software delivery as a significant risk.

An exposure analysis of this risk would have shown that since the baggage-handling software was on the critical path, any delay would postpone the airport's opening, resulting in financial penalties of $33 million per month. (That carrying cost would have been easily calculable from the beginning.) From there, it would have been an obvious conclusion that moving the software off the critical path was a key mitigation strategy. A few million

dollars spent early in the effort to make an alternative baggage-handling scheme feasible would have saved half a billion dollars when the software project did not complete on time.

At the very end of this book, we list a dozen or so necessary actions that together constitute risk management. As you will see, DIA upper management methodically observed precisely zero of these.

So, Who Blew It?

Since the contractor has already taken so much heat for its failure to deliver DIA's ABHS on time, it seems only fair to mention here that risk management was not entirely the contractor's job. If you agree with our assessment that this was a failure of risk management far more than of software process, then it makes no sense to blame the contractor. In fact, the risk of the $500 million of extra financing cost belonged at the next level up. Responsibility for risk management accrues to whichever party will have to pay the price for risks that are ignored.

In this case, all such costs were eventually paid for by the contracting agency, Denver Airport System, an arm of the city government. Thus, the city of Denver was responsible for managing the financing risk, something it made no discernible effort to do.

INDEX

Acceptance testing, 63, 124, 132
Accountability, 147, 153, 154, 159
  for cost, 149, 155
  of stakeholders, 156
Austin, Rob, 154
Automated Baggage Handling System (ABHS), 22-23, 25-28, 70, 184

Benefit, 63, 64, 109, 146ff., 151, 152-53, 154, 161, 162
  calculation, 149
  expected, 149, 152, 153
  quantification, 147, 148
  realization, 146, 148, 154
  risk and, 9, 63, 137, 160
Boehm, Barry W., 119, 150, 158, 190
Brainstorming, 18, 116
Budget, 5, 135, 147, 155, 165
  constraints, 159
  reserve, 69, 70-71
  time, 102, 147

Can-do, 13, 16, 31, 40, 61-62, 114-26, 171
Can't-do, 13, 31, 115
Capability Maturity Model (CMM), 15, 23
Catastrophe, 115, 118
  brainstorming, 116
  list, 45
Causal risks, 16, 86, 87, 172
Charette, Robert, 12, 183, 184, 185
Client, 30, 33, 39, 43-44, 65, 84, 138, 156
Clifford, William Kingdon, 3-6, 17, 64, 175ff.
Closure metrics, 122, 135, 137, 166, 173
Commit date, 75
Communication breakdown, 114
Conflict, 24, 108, 109, 119, 120, 123, 129
Contingency actions, 74, 165, 173
Contingency planning, 20, 71, 74, 167, 173
Core risks, 91, 100, 101-11, 112, 113, 122, 127, 164, 165, 172
  combined effect of, 111
Cost:
  avoided, 154-55

benefit analysis, 130, 143, 145, 146, 149
expected, 149
reduction, 22, 64, 154
specification, 147
value trade-off, 80
Critical path, 25, 27, 65
Culture, corporate, 21, 29, 35, 40, 42, 76, 113, 114, 117, 171, 172
Cumulative risk diagram, 90, 96, 152

Dataflows, 109-10, 123, 124, 126, 166
Davis, Christine, 154, 155
Deadlines, 5, 48, 53, 56, 57, 77-78
Death-march projects, 160-61
Delivery date, 27, 29, 30, 31, 39, 40, 42, 53ff., 69, 76, 83, 86, 90, 100, 129, 130, 140, 153, 165, 167
buffered, 69
most likely, 53
optimistic, 71, 74
realistic, 71
Denver International Airport (DIA), 22-23, 24-28, 29, 33, 50, 70, 184
Design, 108, 123, 128, 157
blueprint, 132-33, 134, 135
partitioning, 135, 166
Diseconomies of scale, 158
Disincentives, 114-15, 171
Doubt, 5, 42, 151, 175, 177

Early delivery, 76, 158
Early-start option, 138, 140, 141
Earned value running (EVR), 122, 124-27, 129, 135ff., 166, 167
Economies of scale, 158
Efficiency, 12, 129
Effort, 4, 6, 24, 27, 28, 33, 42, 47, 53, 59, 68, 81, 104, 105, 116, 135, 136, 149, 158ff., 166, 173
Elemental requirement components, 134, 166
Estimates, 39, 57-58, 60, 69, 87, 88, 90, 91, 103-4, 109, 124, 141, 156, 164, 166, 172
of schedule, 73, 74, 102
Estimation process, 106
Expectations, 31, 40, 61, 66, 69, 76, 101, 110, 111, 149, 151, 152, 161, 163, 171
Exposure, 66-67, 68, 69, 73, 165, 173
analysis, 20, 27
assessing, 67

Failure, 16, 25, 28, 30, 31, 32, 33, 41, 42, 45, 61, 63, 66, 86, 109, 113, 130, 154, 166
accidental, 5, 176
blame and, 117
partial, 118
False positives, 71, 72, 171
Fear, 12, 32, 115-16, 117
Functionality, 77, 100, 109, 126, 146, 154, 157, 159

Gilb, Tom, 132, 187
Glide path, 137, 167
Goals, 24, 32, 39, 40, 75, 76, 172

Implementers, 129
Incremental delivery, 128, 129, 132ff., 166
Incremental implementation, 77, 78, 131
Incrementalism, 78, 124, 128, 129, 130, 131, 137, 156, 173, 187
benefits of, 128-29, 130, 137
constraints on, 135
mitigation and, 128
overlap of tasks, 135
proactive, 130
reactive, 129-30
value/cost analysis of, 157

value projections and, 159
version plans and, 124
Incremental risk diagram, 90, 96, 152
Innovation, 24

Jackson, Sean, 154

Keen, Peter G.W., 109, 183

Lateness, 25-27, 30, 42, 48, 76, 130, 140
Luck, 4, 46, 47-48, 64, 121
Lying, 38, 40

Management, 24, 27, 34, 39, 44, 104, 117, 129, 145, 149, 171
gutless, 140
malpractice, 17
by objectives, 37
reckless, 34
styles, 37
for success, 26, 35, 39
upper, 26, 28, 172
Managers, 37, 38, 40, 56ff., 60, 61, 63, 65, 76, 79, 102, 104, 130, 131, 140, 141, 147, 153ff., 171
upper, 63, 70, 103
Market window, 148, 153
Maturity, 15-16, 23, 38, 58
McMenamin, Steve, 148, 155
Mean-time-to-failure, 80
Methods, 23
Metrics, 119, 122, 124, 160
closure, 122, 135, 137, 166, 173
Monte Carlo:
sampling, 96, 99
simulator, 98, 99
tools, 153
Motivation, 137
Myopia, selective, 43, 44

Nano-percent (N):
date, 57-59, 74, 88, 165
functionality, 77
performance level, 172
scheduling, 74-75
Negative thinking, 31, 114
Net present value (NPV), 145, 163
Noise, 59, 110

Opportunity, 10, 14, 30, 33, 104, 111
cost, 149
Overcommitment, 79
Oversizing, 102
Overstaffing, 18

Parametric estimator, 87, 91, 165
Parkinsonian scheduling, 37
Piling on, 109, 130
Planning, 15, 19, 39, 48, 64, 72, 82, 101, 105, 121
for risk, 15, 16
Politics, 24, 40, 59, 79, 138, 156
Postmortems, 61-62, 122, 149, 188
Probability analysis, 118
Probability of delivery, 54, 57, 77, 78, 90
Production, 86, 88, 91, 137
Prognosticator, 87
Project:
assumptions, 67, 68, 74, 152, 165
benefits, 143, 146-47
cancellation, 43, 44, 45, 65, 67, 108, 109, 110, 112, 124, 129, 131
fixed-deadline, 77-78
high-value, 149
justification, 161-62
low-value, 149, 161
management, 5, 15, 68, 80, 108
managers, 16, 30, 32, 40, 49, 53, 59, 61, 79, 102, 104, 129, 132, 152
parameters, 111, 165
plan, 16, 48, 60, 69, 70, 74, 101, 102, 129, 165, 173
relative size of, 173

risk, *viii,* 30, 31
review, 116
team, 27, 68, 110
visibility, 137
Promises, 30, 39, 61, 154, 155
Pseudo delivery, 173

Rank-order prioritization, 129, 130, 131, 159, 166
Requirements, 18, 24, 95, 101, 105, 106, 119, 134, 146, 150, 166
inflation, 102, 104-6
setting, 108
time and, 105, 106
Resizing, 104
Responsibility, 19, 20, 21, 67, 68, 114, 177
for measurement, 156
for risk, 33, 152
for risk management, 28, 29, 65, 171
transfers of, 33, 66
for value-projection, 156
Return on investment, 145
Risk, *v, vii,* 7, 11ff., 16ff., 24, 26ff., 31ff., 39-40, 43ff., 51, 58, 59, 61, 63ff., 73, 74, 77, 79ff., 113ff., 118ff., 124, 127ff., 137, 138, 140, 143, 147, 149, 152, 153, 160ff., 172, 173
analysis, 20, 29
assessment, *viii,* 41, 83, 91, 114, 181
aversion to, 30, 32, 33, 147
awareness of, 13, 60, 130
balancing opportunity and, 14
benefits and, 9
census, 19, 67, 73, 79, 164, 172
containment, *viii,* 31, 63, 64, 66, 67, 68, 69, 70, 137, 139
evasion, 63, 64, 77
expiration, 63, 74, 167
identifying, 43, 45, 66, 113, 115, 117, 181
ignoring, 13, 17, 28, 46-47, 79
indicators, 80, 87
list, 30, 40, 43, 44, 45, 46, 61, 62, 63, 64, 65, 68, 79, 122
minimizing, 31
minor, 13, 58
officer, 20
problems and, 17, 44-45, 61, 65
profile, 86
on project, *viii,* 30, 31
quantifying, 67, 91, 113
repository, 67, 122, 181
reserve, 32, 51, 64, 67, 68-69
running away from, *v,* 9, 10, 11, 12, 14
running toward, 9, 11, 14
simulators, 100
tracking, 20, 43, 51
transition, 17-18, 19, 47, 64, 74, 165, 167
triage, 20
unmanaged, 7, 14, 47
unthinkable, 21
Risk diagrams, 55-57, 58-59, 74ff., 80, 82ff., 93, 95, 96, 98, 100, 103, 104, 147, 151ff., 162ff., 172, 173
shapes of, 85-86
Risk-discovery process, 20, 45, 51, 62, 73, 74, 79, 113, 115, 116, 119, 120, 164, 167, 172
brainstorm, 19, 27
meeting, 116
WinWin approach to, 119-20
Risk management, *vii, viii,* 6, 13-14, 15ff., 24, 27, 28, 29ff., 37-40, 42, 44-45, 46, 48, 50, 51, 61, 62, 64, 65-66, 67, 68, 73, 79, 80, 81, 100, 102, 112, 113, 119, 121, 147, 149, 164, 169, 171, 172, 173
activities, 20, 121
case against, 37-41
cost of not performing, 29
downside of, 35, 37

dynamics, 121ff.
as focusing mechanism, 34
justification for, 17
mechanics of, 60ff.
opposition to, 38
philosophy of, 14
prescription for, 73-74, 164-67
problems of, 14
test for, 171-73
tools, *viii,* 40, 91
Risk manager, *vii,* 17, 99, 129, 172
would-be, 42
Risk materialization, 17, 18, 20, 33, 39, 47, 63, 64, 67, 69, 74, 77, 128, 137, 164, 167, 173
indications of, 71, 73
probability, 20, 46, 66, 69, 73, 165
schedule impact of, 73, 164
Risk mitigation, *viii,* 18, 19, 20, 27, 32, 43, 47, 63, 70, 82, 128, 138, 140, 165, 173
actions, 74, 165, 173
cost of, 16, 64, 67, 69-71, 73, 128, 164
mode, 138
plan, 71, 173
Risk model, 87, 88, 91, 92
customization of, 91, 92
Risk-taking, *vii, viii,* 10, 12, 14, 38, 58, 140
aggressive, 29, 149
necessity of, 13
philosophy of, 160
rewarding, 12
RISKOLOGY, *viii,* 91, 99, 100, 101, 106, 107, 109, 111, 119, 162, 165
Rook, Paul, *v,* 31n., 60
Root cause analysis, 118-19, 189

Sampling tool, 95, 98
Scenario-building, 118
Schedule, 6, 13, 21, 25ff., 31ff., 48, 50, 57, 60, 71, 74, 78, 88, 101ff., 130, 141, 149
estimating, 74, 102, 165
expected, 149
flaw, 102-4
gaming of, 76
overrun, 104
reserve, 69, 70-71
slip, 21
uncertainty about, 76
Sensitivity analysis, 149, 156-57
Showstoppers, 67-68, 74, 165
Silves, Mike, 147
Software:
development, 53, 91, 119
engineering, 23, 150
industry, 23, 38, 56, 59, 101, 103, 104, 140, 145, 160, 173
managers, 38, 139, 149, 158
Software Engineering Institute (SEI), 186
Specification, 43, 53, 108, 109, 123, 134, 157
breakdown, 102, 108-10, 147, 166
closure, 109
Spiral development life cycle, 119
Stakeholders, 22, 31, 38, 53, 75-78, 79, 84, 105, 108, 109, 118, 119, 123, 124, 130, 131, 141, 147ff., 153ff., 165, 173
assumptions of, 152
Status meetings, 43, 63
Stretch goal, 31, 48, 75
Success, 14, 24, 26, 31, 35, 39, 48, 79, 86, 119, 154, 158, 175
System, 11, 13, 24, 38, 43-44, 114, 123ff., 129, 141, 145ff., 150ff., 157, 159
construction, 124
projects, 156, 158
value, 149, 151

Testosterone-based decision making, 154
Theory W, 119
Tools, customizing, 43, 91, 92, 99, 165
Transformation rules, 123
Transition indicators, 17-18, 19, 20, 71, 72, 73, 122, 164, 173
Trucker's maxim, 71
Turnover, 18, 24, 102, 106-8

Uncertainty, 6, 23-24, 29ff., 38-39, 40, 42, 53-54, 56ff., 74ff., 82ff., 91ff., 98, 100, 147, 149, 162, 165, 172
  admitting, 42
  analysis of, 31, 87
  bounded, 32, 57
  confronting, 38
  curve, 59
  diagram, 55, 58, 77, 82, 84, 86, 88, 89, 90, 93, 95, 96, 98, 103, 151, 152, 162, 163, 172, 173
  effect of, 92
  fear of expressing, 171
  functional, 76
  overlayer, 87
  pattern, 95
  quantifying, 40, 53, 56
  schedule, 76
  trade-off, 76
Under-performance, 110
Unknowns, 51, 81, 82, 93
Unwritten rules, 114, 115, 116, 117
Urgency assessment, 72
Users, 64, 77, 80, 113, 117, 129, 131, 137, 156, 159
  involvement of, 137

Value, 40, 55, 65, 70, 82, 96, 98, 114, 122, 130, 131, 136, 137, 143, 145, 149, 150, 151-55, 156ff., 166, 172, 173
  assessment, 143, 154, 155, 158, 166, 173
  cost, ratio to, 157, 158, 159
  distribution, 157
  expected, 151, 161
  prediction, 151, 153, 156, 159
  quantification, 145
  rank-ordering of, 173
  software engineering, 150
Version acceptance test (VAT), 127, 132, 134, 135, 136, 137, 167
Version plan, 124, 158, 173
Versions, 77, 86, 125ff., 130ff., 140, 158, 166, 167, 173
Voice-of-the-product metric, 137

Win conditions, 119, 120
Windows of uncertainty, 29, 30, 39, 58-59, 110
WinWin Spiral Process Model, 119, 190
Work breakdown structure (WBS), 132, 133, 135, 136, 166, 173