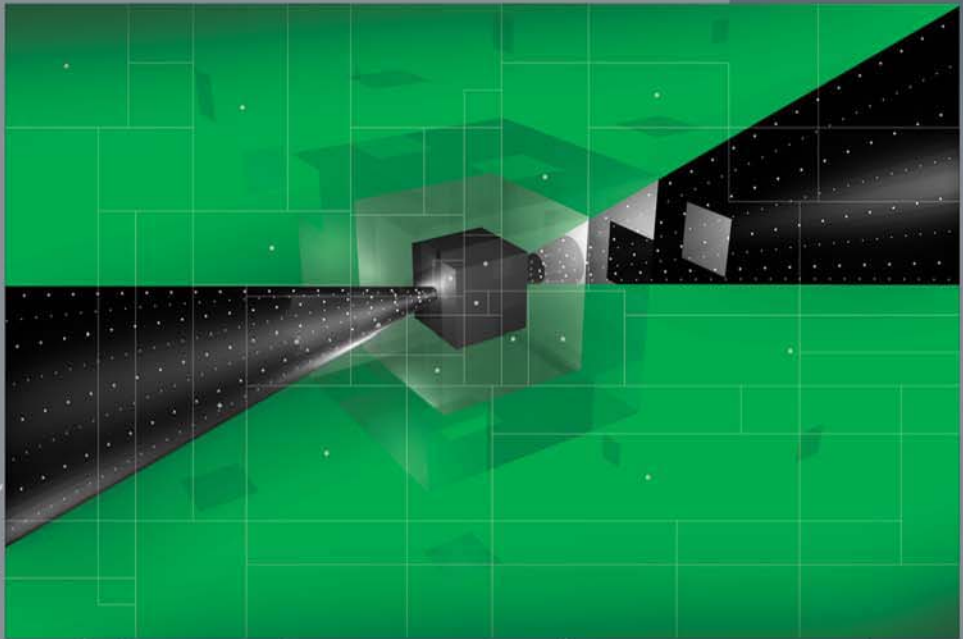


BEST PRACTICES

SOFTWARE ESTIMATION



Demystifying the Black Art

Steve McConnell

Two-time winner of *Software Development* magazine's Jolt Award

PUBLISHED BY
Microsoft Press
A Division of Microsoft Corporation
One Microsoft Way
Redmond, Washington 98052-6399

Copyright © 2006 by Steve McConnell

All rights reserved. No part of the contents of this book may be reproduced or transmitted in any form or by any means without the written permission of the publisher.

Library of Congress Control Number 2005936847

ISBN: 978-0-7356-0535-0

Printed and bound in the United States of America.

7 8 9 10 11 12 13 14 15 QGT 7 6 5 4 3 2

Distributed in Canada by H.B. Fenn and Company Ltd..

A CIP catalogue record for this book is available from the British Library

Microsoft Press books are available through booksellers and distributors worldwide. For further information about international editions, contact your local Microsoft Corporation office or contact Microsoft Press International directly at fax (425) 936-7329. Visit our Web site at www.microsoft.com/mspress. Send comments to mspinput@microsoft.com.

Microsoft, Excel, Microsoft Press, Visual Basic, Windows, and Windows NT are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. Other product and company names mentioned herein may be the trademarks of their respective owners.

The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious. No association with any real company, organization, product, domain name, e-mail address, logo, person, place, or event is intended or should be inferred.

This book expresses the author's views and opinions. The information contained in this book is provided without any express, statutory, or implied warranties. Neither the authors, Microsoft Corporation, nor its resellers or distributors will be held liable for any damages caused or alleged to be caused either directly or indirectly by this book.

Acquisitions Editor: Ben Ryan
Project Editor: Devon Musgrave
Copy Editor: Becka McKay
Indexer: Seth Maislin

Body Part No. X11-82276

Contents at a Glance

Part I Critical Estimation Concepts

1	What Is an “Estimate”?	3
2	How Good an Estimator Are You?	15
3	Value of Accurate Estimates	21
4	Where Does Estimation Error Come From?	33
5	Estimate Influences	55

Part II Fundamental Estimation Techniques

6	Introduction to Estimation Techniques	77
7	Count, Compute, Judge	83
8	Calibration and Historical Data	91
9	Individual Expert Judgment	105
10	Decomposition and Recomposition	113
11	Estimation by Analogy	127
12	Proxy-Based Estimates	135
13	Expert Judgment in Groups	149
14	Software Estimation Tools	157
15	Use of Multiple Approaches	165
16	Flow of Software Estimates on a Well-Estimated Project	171
17	Standardized Estimation Procedures	181

Part III Specific Estimation Challenges

18	Special Issues in Estimating Size	197
19	Special Issues in Estimating Effort	207
20	Special Issues in Estimating Schedule	221
21	Estimating Planning Parameters	233
22	Estimate Presentation Styles	249
23	Politics, Negotiation, and Problem Solving	259
A	Estimation Sanity Check	271
B	Answers to Chapter 2 Quiz, “How Good an Estimator Are You?”	273
C	Software Estimation Tips	275

Table of Contents

Welcome xv
Acknowledgments xxi
List of Equations xxiii
List of Figures xxv

Part I Critical Estimation Concepts

1 What Is an “Estimate”? 3

 1.1 Estimates, Targets, and Commitments 3

 1.2 Relationship Between Estimates and Plans 4

 1.3 Communicating about Estimates, Targets, and Commitments 5

 1.4 Estimates as Probability Statements 6

 1.5 Common Definitions of a “Good” Estimate 9

 1.6 Estimates and Project Control 11

 1.7 Estimation’s Real Purpose 13

 1.8 A Working Definition of a “Good Estimate” 14

 Additional Resources 14

2 How Good an Estimator Are You? 15

 2.1 A Simple Estimation Quiz 15

 2.2 Discussion of Quiz Results 16

 How Confident Is “90% Confident”? 16

 How Wide Should You Make Your Ranges? 18

 Where Does Pressure to Use Narrow Ranges Come From? 18

 How Representative Is This Quiz of Real Software Estimates? 19

3 Value of Accurate Estimates 21

 3.1 Is It Better to Overestimate or Underestimate? 21

 Arguments Against Overestimation 21

 Arguments Against Underestimation 22

 Weighing the Arguments 23

What do you think of this book?
We want to hear from you!

Microsoft is interested in hearing your feedback about this publication so we can continually improve our books and learning resources for you. To participate in a brief online survey, please visit: www.microsoft.com/learning/booksurvey/

3.2 Details on the Software Industry's Estimation Track Record	24
How Late Are the Late Projects?	26
One Company's Experience	26
The Software Industry's Systemic Problem	27
3.3 Benefits of Accurate Estimates	27
3.4 Value of Predictability Compared with Other Desirable Project Attributes	29
3.5 Problems with Common Estimation Techniques	30
Additional Resources	31
4 Where Does Estimation Error Come From?	33
4.1 Sources of Estimation Uncertainty	34
4.2 The Cone of Uncertainty	35
Can You Beat the Cone?	37
The Cone Doesn't Narrow Itself.	38
Accounting for the Cone of Uncertainty in Software Estimates.	39
Relationship Between the Cone of Uncertainty and Commitment	40
The Cone of Uncertainty and Iterative Development	40
4.3 Chaotic Development Processes	41
4.4 Unstable Requirements.	42
Estimating Requirements Growth	43
4.5 Omitted Activities	44
4.6 Unfounded Optimism.	46
4.7 Subjectivity and Bias	47
4.8 Off-the-Cuff Estimates	49
4.9 Unwarranted Precision	51
4.10 Other Sources of Error	52
Additional Resources.	53
5 Estimate Influences	55
5.1 Project Size	55
Why Is This Book Discussing Size in Lines of Code?	56
Diseconomies of Scale	56
When You Can Safely Ignore Diseconomies of Scale.	60
Importance of Diseconomy of Scale in Software Estimation	61
5.2 Kind of Software Being Developed.	61
5.3 Personnel Factors.	63
5.4 Programming Language	64
5.5 Other Project Influences.	65
5.6 Diseconomies of Scale Revisited	70
Additional Resources.	72

Part II Fundamental Estimation Techniques

6	Introduction to Estimation Techniques	77
6.1	Considerations in Choosing Estimation Techniques	77
	What's Being Estimated	77
	Project Size	78
	Software Development Style	78
	Development Stage	80
	Accuracy Possible	80
6.2	Technique Applicability Tables	81
7	Count, Compute, Judge	83
7.1	Count First	84
7.2	What to Count	85
7.3	Use Computation to Convert Counts to Estimates	86
7.4	Use Judgment Only as a Last Resort	88
	Additional Resources	89
8	Calibration and Historical Data	91
8.1	Improved Accuracy and Other Benefits of Historical Data	91
	Accounts for Organizational Influences	92
	Avoids Subjectivity and Unfounded Optimism	93
	Reduces Estimation Politics	93
8.2	Data to Collect	95
	Issues Related to Size Measures	95
	Issues Related to Effort Measures	96
	Issues Related to Calendar Time Measures	97
	Issues Related to Defect Measures	97
	Other Data Collection Issues	98
8.3	How to Calibrate	98
8.4	Using Project Data to Refine Your Estimates	99
8.5	Calibration with Industry Average Data	100
8.6	Summary	102
	Additional Resources	102

9	Individual Expert Judgment	105
9.1	Structured Expert Judgment	106
	Who Creates the Estimates?	106
	Granularity	106
	Use of Ranges	107
	Formulas	108
	Checklists	110
9.2	Compare Estimates to Actuals	110
	Additional Resources	112
10	Decomposition and Recomposition	113
10.1	Calculating an Accurate Overall Expected Case	113
	The Law of Large Numbers	115
	How Small Should the Estimated Pieces Be?	116
10.2	Decomposition via an Activity-Based Work Breakdown Structure	117
10.3	Hazards of Adding Up Best Case and Worst Case Estimates	118
	Warning: Math Ahead!	119
	What Went Wrong?	119
10.4	Creating Meaningful Overall Best Case and Worst Case Estimates	120
	Computing Aggregate Best and Worst Cases for Small Numbers of Tasks (Simple Standard Deviation Formula)	121
	Computing Aggregate Best and Worst Cases for Large Numbers of Tasks (Complex Standard Deviation Formula)	122
	Creating the Aggregate Best and Worst Case Estimates	124
	Cautions About Percentage Confident Estimates	126
	Additional Resources	126
11	Estimation by Analogy	127
11.1	Basic Approach to Estimating by Analogy	127
	Step 1: Get Detailed Size, Effort, and Cost Results for a Similar Previous Project	128
	Step 2: Compare the Size of the New Project to a Similar Past Project	129
	Step 3: Build Up the Estimate for the New Project's Size as a Percentage of the Old Project's Size	130
	Step 4: Create an Effort Estimate from the Size of the New Project Compared to the Previous Project	131
	Step 5: Check for Consistent Assumptions Across the Old and New Projects	131

11.2	Comments on Uncertainty in the Triad Estimate	132
	Estimation Uncertainty, Plans, and Commitments	133
12	Proxy-Based Estimates	135
12.1	Fuzzy Logic	136
	How to Get the Average Size Numbers	136
	How to Classify New Functionality	137
	How Not to Use Fuzzy Logic	137
	Extensions of Fuzzy Logic	138
12.2	Standard Components	138
	Using Standard Components with Percentiles	140
	Limitations of Standard Components	141
12.3	Story Points	142
	Cautions About Ratings Scales	143
12.4	T-Shirt Sizing	145
12.5	Other Uses of Proxy-Based Techniques	147
12.6	Additional Resources	147
13	Expert Judgment in Groups	149
13.1	Group Reviews	149
13.2	Wideband Delphi	150
	Effectiveness of Wideband Delphi	152
	"The Truth Is Out There"	154
	When to Use Wideband Delphi	154
	Additional Resources	155
14	Software Estimation Tools	157
14.1	Things You Can Do with Tools That You Can't Do Manually	157
14.2	Data You'll Need to Calibrate the Tools	162
14.3	One Thing You Shouldn't Do with a Tool Any More than You Should Do Otherwise	162
14.4	Summary of Available Tools	163
	Additional Resources	164
15	Use of Multiple Approaches	165
	Additional Resources	169

16	Flow of Software Estimates on a Well-Estimated Project	171
16.1	Flow of an Individual Estimate on a Poorly Estimated Project	171
16.2	Flow of an Individual Estimate on a Well-Estimated Project	172
16.3	Chronological Estimation Flow for an Entire Project	173
	Estimation Flow for Large Projects	174
	Estimation Flow for Small Projects	175
16.4	Estimate Refinement	175
16.5	How to Present Reestimation to Other Project Stakeholders	176
	When to Present the Reestimates	177
	What If Your Management Won't Let You Reestimate?	178
16.6	A View of a Well-Estimated Project	179
17	Standardized Estimation Procedures	181
17.1	Usual Elements of a Standardized Procedure	181
17.2	Fitting Estimation into a Stage-Gate Process	182
17.3	An Example of a Standardized Estimation Procedure for Sequential Projects	185
17.4	An Example of a Standardized Estimation Procedure for Iterative Projects . .	188
17.5	An Example of a Standardized Estimation Procedure from an Advanced Organization	190
17.6	Improving Your Standardized Procedure	192
	Additional Resources	193
 Part III Specific Estimation Challenges		
18	Special Issues in Estimating Size	197
18.1	Challenges with Estimating Size	197
	Role of Lines of Code in Size Estimation	198
18.2	Function-Point Estimation	200
	Converting from Function Points to Lines of Code	202
18.3	Simplified Function-Point Techniques	203
	The Dutch Method	203
	GUI Elements	204
18.4	Summary of Techniques for Estimating Size	205
	Additional Resources	206

19	Special Issues in Estimating Effort.	207
	19.1 Influences on Effort	207
	19.2 Computing Effort from Size	209
	Computing Effort Estimates by Using Informal Comparison to Past Projects	209
	What Kinds of Effort Are Included in This Estimate?	210
	19.3 Computing Effort Estimates by Using the Science of Estimation	210
	19.4 Industry-Average Effort Graphs	210
	19.5 ISBSG Method	216
	19.6 Comparing Effort Estimates	218
	Additional Resources	219
20	Special Issues in Estimating Schedule.	221
	20.1 The Basic Schedule Equation	221
	20.2 Computing Schedule by Using Informal Comparisons to Past Projects	223
	20.3 Jones's First-Order Estimation Practice	224
	20.4 Computing a Schedule Estimate by Using the Science of Estimation	225
	20.5 Schedule Compression and the Shortest Possible Schedule	226
	20.6 Tradeoffs Between Schedule and Effort	228
	Schedule Compression and Team Size	229
	20.7 Schedule Estimation and Staffing Constraints	230
	20.8 Comparison of Results from Different Methods	231
	Additional Resources	232
21	Estimating Planning Parameters	233
	21.1 Estimating Activity Breakdown on a Project	233
	Estimating Allocation of Effort to Different Technical Activities	233
	Estimating Requirements Effort	234
	Estimating Management Effort	235
	Estimating Total Activity	235
	Adjustments Due to Project Type	236
	Example of Allocating Effort to Activities	237
	Developer-to-Tester Ratios	237
	21.2 Estimating Schedule for Different Activities	238
	21.3 Converting Estimated Effort (Ideal Effort) to Planned Effort	239
	21.4 Cost Estimates	241
	Overtime	241

	Is the Project Cost Based on Direct Cost, Fully Burdened Cost, or Some Other Variation?	241
	Other Direct Costs	241
	21.5 Estimating Defect Production and Removal	241
	Estimating Defect Removal	242
	An Example of Estimating Defect-Removal Efficiency	243
	21.6 Estimating Risk and Contingency Buffers	245
	21.7 Other Rules of Thumb	247
	21.8 Additional Resources	247
22	Estimate Presentation Styles	249
	22.1 Communicating Estimate Assumptions	249
	22.2 Expressing Uncertainty	251
	Plus-or-Minus Qualifiers	251
	Risk Quantification	251
	Confidence Factors	252
	Case-Based Estimates	254
	Coarse Dates and Time Periods	255
	22.3 Using Ranges (of Any Kind)	256
	Usefulness of Estimates Presented as Ranges	256
	Ranges and Commitments	257
	Additional Resources	257
23	Politics, Negotiation, and Problem Solving	259
	23.1 Attributes of Executives	259
	23.2 Political Influences on Estimates	260
	External Constraints	260
	Budgeting and Dates	261
	Negotiating an Estimate vs. Negotiating a Commitment	261
	What to Do if Your Estimate Doesn't Get Accepted	262
	Responsibility of Technical Staff to Educate Nontechnical Stakeholders	262
	23.3 Problem Solving and Principled Negotiation	263
	A Problem-Solving Approach to Negotiation	264
	Separate the People from the Problem	264
	Focus on Interests, Not Positions	265
	Invent Options for Mutual Gain	266
	Insist on Using Objective Criteria	268
	Additional Resources	270

A	Estimation Sanity Check	271
B	Answers to Chapter 2 Quiz, “How Good an Estimator Are You?”	273
C	Software Estimation Tips	275
	Bibliography	287
	Index.	295

What do you think of this book?
We want to hear from you!

Microsoft is interested in hearing your feedback about this publication so we can continually improve our books and learning resources for you. To participate in a brief online survey, please visit: www.microsoft.com/learning/booksurvey/

Welcome

The most unsuccessful three years in the education of cost estimators appears to be fifth-grade arithmetic.

—Norman R. Augustine

Software estimation is not hard. Experts have been researching and writing about software estimation for four decades, and they have developed numerous techniques that support accurate software estimates. Creating accurate estimates is straightforward—once you understand how to create them. But not all estimation practices are intuitively obvious, and even smart people won’t discover all the good practices on their own. The fact that someone is an expert developer doesn’t make that person an expert estimator.

Numerous aspects of estimation are not what they seem. Many so-called estimation problems arise from misunderstanding what an “estimate” is or blurring other similar-but-not-identical concepts with estimation. Some estimation practices that seem intuitively useful don’t produce accurate results. Complex formulas sometimes do more harm than good, and some deceptively simple practices produce surprisingly accurate results.

This book distills four decades of research and even more decades of hands-on experience to help developers, leads, testers, and managers become effective estimators. Learning about software estimation turns out to be generally useful because the influences that affect software estimates are the influences that affect software development itself.

Art vs. Science of Software Estimation

Software estimation research is currently focused on improving estimation techniques so that sophisticated organizations can achieve project results within $\pm 5\%$ of estimated results instead of within $\pm 10\%$. These techniques are mathematically intensive. Understanding them requires a strong math background and concentrated study. Using them requires number crunching well beyond what you can do on your hand calculator. These techniques work best when embodied in commercial software estimation tools. I refer to this set of practices as the *science of estimation*.

Meanwhile, the typical software organization is not struggling to improve its estimates from $\pm 10\%$ to $\pm 5\%$ accuracy. The typical software organization is struggling to avoid estimates that are incorrect by 100% or more. (The reasons for this are manifold and will be discussed in detail in Chapters 3 and 4.)

Our natural tendency is to believe that complex formulas like this:

$$\text{Effort} = 2.94 * (\text{KSLOC})^{[0.91 + 0.01 * \sum_{j=1}^5 SF_j]} * \prod_{i=1}^{17} EM_i$$

will always produce more accurate results than simple formulas like this:

$$\text{Effort} = \text{NumberOfRequirements} * \text{AverageEffortPerRequirement}$$

But complex formulas aren't necessarily better. Software projects are influenced by numerous factors that undermine many of the assumptions contained in the complex formulas of the science of estimation. Those dynamics will be explained later in this book. Moreover, most software practitioners have neither the time nor the inclination to learn the intensive math required to understand the science of estimation.

Consequently, this book emphasizes rules of thumb, procedures, and simple formulas that are highly effective and understandable to practicing software professionals. These techniques will not produce estimates that are accurate to within $\pm 5\%$, but they will reduce estimation error to about 25% or less, which turns out to be about as useful as most projects need, anyway. I call this set of techniques the *art of estimation*.

This book draws from both the art and science of software estimation, but its focus is on software estimation as an art.

Why This Book Was Written and Who It Is For

The literature on software estimation is widely scattered. Researchers have published hundreds of articles, and many of them are useful. But the typical practitioner doesn't have time to track down dozens of papers from obscure technical journals. A few previous books have described the science of estimation. Those books are 800–1000 pages long, require a good math background, and are targeted mainly at professional estimators—consultants or specialists who estimate large projects and do so frequently.

I wrote this book for developers, leads, testers, and managers who need to create estimates occasionally as one of their many job responsibilities. I believe that most practitioners want to improve the accuracy of their estimates but don't have the time to obtain a Ph.D. in software estimation. These practitioners struggle with practical issues like how to deal with the politics that surround the estimate, how to present an estimate so that it will actually be accepted, and how to avoid having someone change your estimate arbitrarily. If you are in this category, this book was written for you.

The techniques in this book apply to Internet and intranet development, embedded software, shrink-wrapped software, business systems, new development, legacy systems, large projects, small projects—essentially, to estimates for all kinds of software.

Key Benefits Of This Book

By focusing on the art of estimation, this book provides numerous important estimation insights:

- What an “estimate” is. (You might think you already know what an estimate is, but common usages of the term are inaccurate in ways that undermine effective estimation.)
- The specific factors that have made your past estimates less accurate than they could have been.
- Ways to distinguish a good estimate from a poor one.
- Numerous techniques that will allow *you personally* to create good estimates.
- Several techniques you can use to help *other people on your team* create good estimates.
- Ways that *your organization* can create good estimates. (There are important differences between personal techniques, group techniques, and organizational techniques.)
- Estimation approaches that work on agile projects, and approaches that work on traditional, sequential (plan-driven) projects.
- Estimation approaches that work on small projects and approaches that work on large projects.
- How to navigate the shark-infested political waters that often surround software estimation.

In addition to gaining a better understanding of estimation concepts, the practices in this book will help you estimate numerous specific attributes of software projects, including:

- New development work, including schedule, effort, and cost
- Schedule, effort, and cost of legacy systems work
- How many features you can deliver within a specific development iteration
- The amount of functionality you can deliver for a whole project when schedule and team size are fixed
- Proportions of different software development activities needed, including how much management work, requirements, construction, testing, and defect correction will be needed
- Planning parameters, such as tradeoffs between cost and schedule, best team size, amount of contingency buffer, ratio of developers to testers, and so on

- Quality parameters, including time needed for defect correction work, defects that will remain in your software at release time, and other factors
- Practically anything else you want to estimate

In many cases, you'll be able to put this book's practices to use right away.

Most practitioners will not need to go any further than the concepts described in this book. But understanding the concepts in this book will lay enough groundwork that you'll be able to graduate to more mathematically intensive approaches later on, if you want to.

What This Book Is Not About

This book is not about how to estimate the very largest projects—more than 1 million lines of code, or more than 100 staff years. Very large projects should be estimated by professional estimators who have read the dozens of obscure journal articles, who have studied the 800–1000-page books, who are familiar with commercial estimation software, and who are as skilled in both the art and science of estimation.

Where to Start

Where you start will depend on what you want to get out of the book.

If you bought this book because you need to create an estimate right now... Begin with Chapter 1 (“What Is an “Estimate”?”), and then move to Chapter 7 (“Count, Compute, Judge”) and Chapter 8 (“Calibration and Historical Data”). After that, skim the tips in Chapters 10–20 to find the techniques that will be the most immediately useful to you. By the way, this book's tips are highlighted in the text and numbered, and all of the tips—118 total—are also collected in Appendix C, “Software Estimation Tips.”

If you want to improve your personal estimation skills, if you want to improve your organization's estimation track record, or if you're looking for a better understanding of software estimation in general... You can read the whole book. If you like to understand general principles before you dive into the details, read the book in order. If you like to see the details first and then draw general conclusions from the details, you can start with Chapter 1, read Chapters 7 through 23, and then go back and read the earlier chapters that you skipped.

Bellevue, Washington
New Year's Day, 2006

Microsoft Press Support

Every effort has been made to ensure the accuracy of this book. Microsoft Press provides corrections for books through the World Wide Web at the following address:

<http://www.microsoft.com/learning/support/books/>

To connect directly to the Microsoft Press Knowledge Base and enter a query regarding a question or issue that you may have, go to:

<http://www.microsoft.com/mspress/support/search.asp>

If you have comments, questions, or ideas regarding this book, please send them to Microsoft Press using either of the following methods:

Postal Mail:

*Microsoft Press
Attn: Software Estimation Editor
One Microsoft Way
Redmond, WA 98052-6399*

E-Mail:

mspinput@microsoft.com

Acknowledgments

I continue to be amazed at the many ways the Internet supports high-quality work. My first book's manuscript was reviewed almost entirely by people who lived within 50 miles of me. This book's manuscript included reviewers from Argentina, Australia, Canada, Denmark, England, Germany, Iceland, The Netherlands, Northern Ireland, Japan, Scotland, Spain, and the United States. The book has benefited enormously from these reviews.

Thanks first to the people who contributed review comments on significant portions of the book: Fernando Berzal, Steven Black, David E. Burgess, Stella M. Burns, Gavin Burrows, Dale Campbell, Robert A. Clinkenbeard, Bob Corrick, Brian Donaldson, Jason Hills, William Horn, Carl J. Krzystofczyk, Jeffrey D. Moser, Thomas Oswald, Alan M. Pinder, Jon Price, Kathy Rhode, Simon Robbie, Edmund Schweppe, Gerald Simon, Creig R. Smith, Linda Taylor, and Bernd Viefhues.

Thanks also to the people who reviewed selected portions of the book: Lisa M. Adams, Hákon Ágústsson, Bryon Baker, Tina Coleman, Chris Crawford, Dominic Cronin, Jerry Deville, Conrado Estol, Eric Freeman, Hideo Fukumori, C. Dale Hildebrandt, Barbara Hitchings, Jim Holmes, Rick Hower, Kevin Hutchison, Finnur Hrafn Jonsson, Aaron Kiander, Mehmet Kerem Kızıltunç, Selimir Kustudic, Molly J. Mahai, Steve Mattingly, Joe Nicholas, Al Noel, David O'Donoghue, Sheldon Porcina, David J. Preston, Daniel Read, David Spokane, Janco Tanis, Ben Tilly, and Wendy Wilhelm.

I'd especially like to acknowledge Construx's estimation seminar instructors. After years of stimulating discussions, it's often impossible to tell which ideas originated with me and which originated with them. Thanks to Earl Beede, Gregg Boer, Matt Peloquin, Pamela Perrott, and Steve Tockey.

This book focuses on estimation as an art, and this book's simplifications were made possible by researchers who have spent decades clarifying estimation as a science. My heartfelt appreciation to three of the giants of estimation science: Barry Boehm, Capers Jones, and Lawrence Putnam.

Working with Devon Musgrave, project editor for this book, has once again been a privilege. Thanks, Devon! Becka McKay, assistant editor, also improved my original manuscript in countless ways. Thanks also to the rest of the Microsoft Press staff, including Patricia Bradbury, Carl Diltz, Tracey Freel, Jessie Good, Patricia Masserman, Joel Panchot, and Sandi Resnick. And thanks to indexer Seth Maislin.

Thanks finally to my wife, Ashlie, who is—in my estimation—the best life partner I could ever hope for.

Equations

Equation #1	Program Evaluation and Review Technique (PERT) Formula	109
Equation #2	Pessimistic PERT Formula	109
Equation #3	Magnitude of Relative Error (MRE) Formula	110
Equation #4	Simple Standard Deviation Formula	121
Equation #5	Complex Standard Deviation Formula	122
Equation #6	Modified Complex Standard Deviation Formula	124
Equation #7	PERT Formula for Estimating Number of Components	139
Equation #8	Dutch Method's Indicative Function Point Count Formula	203
Equation #9	ISBSG Effort Formula for General Projects	216
Equation #10	ISBSG Effort Formula for Mainframe Projects	216
Equation #11	ISBSG Effort Formula for Mid-Range Projects	217
Equation #12	ISBSG Effort Formula for Desktop Projects	217
Equation #13	ISBSG Effort Formula for Third Generation Projects	217
Equation #14	ISBSG Effort Formula for Fourth Generation Projects	217
Equation #15	ISBSG Effort Formula for Enhancement Projects	217
Equation #16	ISBSG Effort Formula for New Development Projects	217
Equation #17	The Basic Schedule Equation	221
Equation #18	Informal Comparison to Past Projects Formula	223

Figures

- Figure 1-1** Single-point estimates assume 100% probability of the actual outcome equaling the planned outcome. This isn't realistic. 6
- Figure 1-2** A common assumption is that software project outcomes follow a bell curve. This assumption is incorrect because there are limits to how efficiently a project team can complete any given amount of work. 7
- Figure 1-3** An accurate depiction of possible software project outcomes. There is a limit to how well a project can go but no limit to how many problems can occur. 8
- Figure 1-4** The probability of a software project delivering on or before a particular date (or less than or equal to a specific cost or level of effort). 8
- Figure 1-5** All single-point estimates are associated with a probability, explicitly or implicitly. 9
- Figure 1-6** Improvement in estimation of a set of U.S. Air Force projects. The predictability of the projects improved dramatically as the organizations moved toward higher CMM levels. 10
- Figure 1-7** Improvement in estimation at the Boeing Company. As with the U.S. Air Force projects, the predictability of the projects improved dramatically at higher CMM levels. 10
- Figure 1-8** Schlumberger improved its estimation accuracy from an average overrun of 35 weeks to an average underrun of 1 week. 11
- Figure 1-9** Projects change significantly from inception to delivery. Changes are usually significant enough that the project delivered is not the same as the project that was estimated. Nonetheless, if the outcome is similar to the estimate, we say the project met its estimate. 12
- Figure 2-1** Results from administering the "How Good an Estimator Are You?" quiz. Most quiz-takers get 1–3 answers correct. 17
- Figure 3-1** The penalties for underestimation are more severe than the penalties for overestimation, so, if you can't estimate with complete accuracy, try to err on the side of overestimation rather than underestimation. 24
- Figure 3-2** Project outcomes reported in The Standish Group's Chaos report have fluctuated year to year. About three quarters of all software projects are delivered late or fail outright. 25
- Figure 3-3** Estimation results from one organization. General industry data suggests that this company's estimates being about 100% low is typical. Data used by permission. 26

- Figure 3-4** When given the option of a shorter average schedule with higher variability or a longer average schedule with lower variability, most businesses will choose the second option. 30
- Figure 4-1** The Cone of Uncertainty based on common project milestones. 36
- Figure 4-2** The Cone of Uncertainty based on calendar time. The Cone narrows much more quickly than would appear from the previous depiction in Figure 4-1. 37
- Figure 4-3** If a project is not well controlled or well estimated, you can end up with a Cloud of Uncertainty that contains even more estimation error than that represented by the Cone. 38
- Figure 4-4** The Cone of Uncertainty doesn't narrow itself. You narrow the Cone by making decisions that remove sources of variability from the project. Some of these decisions are about what the project will deliver; some are about what the project will *not* deliver. If these decisions change later, the Cone will widen. 39
- Figure 4-5** A Cone of Uncertainty that allows for requirements increases over the course of the project. 43
- Figure 4-6** Example of variations in estimates when numerous adjustment factors are present. The more adjustment factors an estimation method provides, the more opportunity there is for subjectivity to creep into the estimate. 48
- Figure 4-7** Example of low variation in estimates resulting from a small number of adjustment factors. (The scales of the two graphs are different, but they are directly comparable when you account for the difference in the average values on the two graphs.) 49
- Figure 4-8** Average error from off-the-cuff estimates vs. reviewed estimates. 50
- Figure 5-1** Growth in effort for a typical business-systems project. The specific numbers are meaningful only for the average business-systems project. The general dynamic applies to software projects of all kinds. 56
- Figure 5-2** The number of communication paths on a project increases proportionally to the square of the number of people on the team. 57
- Figure 5-3** Diseconomy of scale for a typical business-systems project ranging from 10,000 to 100,000 lines of code. 58
- Figure 5-4** Diseconomy of scale for projects with greater size differences and the worst-case diseconomy of scale. 59

- Figure 5-5** Differences between ratio-based estimates and estimates based on diseconomy of scale will be minimal for projects within a similar size range. 61
- Figure 5-6** Effect of personnel factors on project effort. Depending on the strength or weakness in each factor, the project results can vary by the amount indicated—that is, a project with the worst requirements analysts would require 42% more effort than nominal, whereas a project with the best analysts would require 29% less effort than nominal. 63
- Figure 5-7** Cocomo II factors arranged in order of significance. The relative lengths of the bars represent the sensitivity of the estimate to the different factors. 67
- Figure 5-8** Cocomo II factors arranged by potential to increase total effort (gray bars) and potential to decrease total effort (blue bars). 68
- Figure 5-9** Cocomo II factors with diseconomy of scale factors highlighted in blue. Project size is 100,000 LOC. 71
- Figure 5-10** Cocomo II factors with diseconomy of scale factors highlighted in blue. Project size is 5,000,000 LOC. 72
- Figure 8-1** An example of estimated outcomes for an estimate calibrated using industry-average data. Total variation in the effort estimates is about a factor of 10 (from about 25 staff months to about 250 staff months). 100
- Figure 8-2** An estimate calibrated using historical productivity data. The effort estimates vary by only about a factor of 4 (from about 30 staff months to about 120 staff months). 101
- Figure 10-1** Software projects tend to progress from large-grain focus at the beginning to fine-grain focus at the end. This progression supports increasing the use of estimation by decomposition as a project progresses. 116
- Figure 13-1** A simple review of individually created estimates significantly improves the accuracy of the estimates. 150
- Figure 13-2** A Wideband Delphi estimating form. 151
- Figure 13-3** A Wideband Delphi estimating form after three rounds of estimates. 152
- Figure 13-4** Estimation accuracy of simple averaging compared to Wideband Delphi estimation. Wideband Delphi reduces estimation error in about two-thirds of cases. 153

- Figure 13-5** Wideband Delphi when applied to terrible initial estimates. In this data set, Wideband Delphi improved results in 8 out of 10 cases. 153
- Figure 13-6** In about one-third of cases, Wideband Delphi helps groups that don't initially include the correct answer to move outside their initial range and closer to the correct answer. 154
- Figure 14-1** A tool-generated simulation of 1,000 project outcomes. Output from Construx Estimate. 158
- Figure 14-2** Example of probable project outcomes based on output from estimation software. 159
- Figure 14-3** In this simulation, only 8 of the 1,000 outcomes fall within the desired combination of cost and schedule. 161
- Figure 14-4** Calculated effect of shortening or lengthening a schedule. 161
- Figure 15-1** An example of multiple estimates for a software project. 168
- Figure 16-1** Estimation on a poorly estimated project. Neither the inputs nor the process are well defined, and the inputs, process, and outputs are all open to debate. 172
- Figure 16-2** Estimation on a well-estimated project. The inputs and process are well defined. The process and outputs are not subject to debate; however, the inputs are subject to iteration until acceptable outputs are obtained. 172
- Figure 16-3** Flow of a single estimate on a well-estimated project. Effort, schedule, cost, and features that can be delivered are all computed from the size estimate. 173
- Figure 16-4** Summary of applicability of different estimation techniques by kind of project and project phase. 174
- Figure 16-5** A well-estimated project. The single-point estimates miss the mark, but the ranges all include the eventual outcome. 179
- Figure 16-6** A poorly estimated project. The project is uniformly underestimated, and the estimation ranges are too narrow to encompass the eventual outcome. 180
- Figure 17-1** A typical stage-gate product development life cycle. 182
- Figure 19-1** Industry-average effort for real-time projects. 211
- Figure 19-2** Industry-average effort for embedded systems projects. 212
- Figure 19-3** Industry-average effort for telecommunications projects. 212
- Figure 19-4** Industry-average effort for systems software and driver projects. 213

Figure 19-5	Industry-average effort for scientific systems and engineering research projects. 213
Figure 19-6	Industry-average effort for shrink-wrap and packaged software projects. 214
Figure 19-7	Industry-average effort for public internet systems projects. 214
Figure 19-8	Industry-average effort for internal intranet projects. 215
Figure 19-9	Industry-average effort for business systems projects. 215
Figure 19-10	Ranges of estimates derived by using the methods discussed in this chapter. The relative dot sizes and line thicknesses represent the weight I would give each of the estimation techniques in this case. 219
Figure 20-1	The Cone of Uncertainty, including schedule adjustment numbers on the right axis. The schedule variability is much lower than the scope variability because schedule is a cube-root function of scope. 222
Figure 20-2	The effects of compressing or extending a nominal schedule and the Impossible Zone. All researchers have found that there is a maximum degree to which a schedule can be compressed. 226
Figure 20-3	Relationship between team size, schedule, and effort for business-systems projects of about 57,000 lines of code. For team sizes greater than 5 to 7 people, effort and schedule both increase. 229
Figure 20-4	Ranges of schedule estimates produced by the methods discussed in this chapter. The relative dot sizes and line thicknesses represent the weights I would give each of these estimates. Looking at all the estimates, including those that aren't well founded, hides the real convergence among these estimates. 231
Figure 20-5	Ranges of schedule estimates produced by the most accurate methods. Once the estimates produced by overly generic methods are eliminated, the convergence of the estimates becomes clear. 232
Figure 22-1	Example of documenting estimate assumptions. 250
Figure 22-2	Example of presenting percentage-confident estimates in a form that's more visually appealing than a table. 253
Figure 22-3	Example of presenting case-based estimates in a visual form. 254

What Is an “Estimate”?

It is very difficult to make a vigorous, plausible, and job-risking defense of an estimate that is derived by no quantitative method, supported by little data, and certified chiefly by the hunches of the managers.

—Fred Brooks

You might think you already know what an estimate is. My goal by the end of this chapter is to convince you that an estimate is different from what most people think. A *good* estimate is even more different.

Here is a dictionary definition of *estimate*: 1. A tentative evaluation or rough calculation. 2. A preliminary calculation of the cost of a project. 3. A judgment based upon one’s impressions; opinion. (Source: *The American Heritage Dictionary*, Second College Edition, 1985.)

Does this sound like what you are asked for when you’re asked for an estimate? Are you asked for a *tentative* or *preliminary* calculation—that is, do you expect that you can change your mind later?

Probably not. When executives ask for an “estimate,” they’re often asking for a commitment or for a plan to meet a target. The distinctions between estimates, targets, and commitments are critical to understanding what an estimate is, what an estimate is not, and how to make your estimates better.

1.1 Estimates, Targets, and Commitments

Strictly speaking, the dictionary definition of *estimate* is correct: an estimate is a prediction of how long a project will take or how much it will cost. But estimation on software projects interplays with business targets, commitments, and control.

A *target* is a statement of a desirable business objective. Examples include the following:

- “We need to have Version 2.1 ready to demonstrate at a trade show in May.”
- “We need to have this release stabilized in time for the holiday sales cycle.”
- “These functions need to be completed by July 1 so that we’ll be in compliance with government regulations.”
- “We must limit the cost of the next release to \$2 million, because that’s the maximum budget we have for that release.”

Businesses have important reasons to establish targets independent of software estimates. But the fact that a target is desirable or even mandatory does not necessarily mean that it is achievable.

While a target is a description of a desirable business objective, a *commitment* is a promise to deliver defined functionality at a specific level of quality by a certain date. A commitment can be the same as the estimate, or it can be more aggressive or more conservative than the estimate. In other words, do not assume that the commitment has to be the same as the estimate; it doesn't.

Tip #1

Distinguish between estimates, targets, and commitments.

1.2 Relationship Between Estimates and Plans

Estimation and planning are related topics, but estimation is not planning, and planning is not estimation. Estimation should be treated as an unbiased, analytical process; planning should be treated as a biased, goal-seeking process. With estimation, it's hazardous to want the estimate to come out to any particular answer. The goal is accuracy; the goal is not to seek a particular result. But the goal of planning is to seek a particular result. We deliberately (and appropriately) bias our plans to achieve specific outcomes. We plan specific means to reach a specific end.

Estimates form the foundation for the plans, but the plans don't have to be the same as the estimates. If the estimates are dramatically different from the targets, the project plans will need to recognize that gap and account for a high level of risk. If the estimates are close to the targets, then the plans can assume less risk.

Both estimation and planning are important, but the fundamental differences between the two activities mean that combining the two tends to lead to poor estimates *and* poor plans. The presence of a strong planning target can lead to substitution of the target for an analytically derived estimate; project members might even refer to the target as an "estimate," giving it a halo of objectivity that it doesn't deserve.

Here are examples of planning considerations that depend in part on accurate estimates:

- Creating a detailed schedule
- Identifying a project's critical path
- Creating a complete work breakdown structure
- Prioritizing functionality for delivery
- Breaking a project into iterations

Accurate estimates support better work in each of these areas (and Chapter 21, "Estimating Planning Parameters," goes into more detail on these topics).

1.3 Communicating about Estimates, Targets, and Commitments

One implication of the close and sometimes confusing relationship between estimation and planning is that project stakeholders sometimes miscommunicate about these activities. Here’s an example of a typical miscommunication:

EXECUTIVE: How long do you think this project will take? We need to have this software ready in 3 months for a trade show. I can’t give you any more team members, so you’ll have to do the work with your current staff. Here’s a list of the features we’ll need.

PROJECT LEAD: OK, let me crunch some numbers, and get back to you.

Later...

PROJECT LEAD: We’ve estimated the project will take 5 months.

EXECUTIVE: Five months!? Didn’t you hear me? I said we needed to have this software ready in 3 months for a trade show!

In this interaction, the project lead will probably walk away thinking that the executive is irrational, because he is asking for the team to deliver 5 months’ worth of functionality in 3 months. The executive will walk away thinking that the project lead doesn’t “get” the business reality, because he doesn’t understand how important it is to be ready for the trade show in 3 months.

Note in this example that the executive was not really asking for an estimate; he was asking the project lead to come up with a *plan* to hit a *target*. Most executives don’t have the technical background that would allow them to make fine distinctions between estimates, targets, commitments, and plans. So it becomes the technical leader’s responsibility to translate the executive’s request into more specific technical terms.

Here’s a more productive way that the interaction could go:

EXECUTIVE: How long do you think this project will take? We need to have this software ready in 3 months for a trade show. I can’t give you any more team members, so you’ll have to do the work with your current staff. Here’s a list of the features we’ll need.

PROJECT LEAD: Let me make sure I understand what you’re asking for. Is it more important for us to deliver 100% of these features, or is it more important to have something ready for the trade show?

EXECUTIVE: We have to have something ready for the trade show. We'd like to have 100% of those features if possible.

PROJECT LEAD: I want to be sure I follow through on your priorities as best I can. If it turns out that we can't deliver 100% of the features by the trade show, should we be ready to ship what we've got at trade show time, or should we plan to slip the ship date beyond the trade show?

EXECUTIVE: We have to have something for the trade show, so if push comes to shove, we have to ship something, even if it isn't 100% of what we want.

PROJECT LEAD: OK, I'll come up with a plan for delivering as many features as we can in the next 3 months.

Tip #2

When you're asked to provide an estimate, determine whether you're supposed to be estimating or figuring out how to hit a target.

1.4 Estimates as Probability Statements

If three-quarters of software projects overrun their estimates, the odds of any given software project completing on time and within budget are not 100%. Once we recognize that the odds of on-time completion are not 100%, an obvious question arises: "If the odds aren't 100%, what are they?" This is one of the central questions of software estimation.

Software estimates are routinely presented as single-point numbers, such as "This project will take 14 weeks." Such simplistic single-point estimates are meaningless because they don't include any indication of the probability associated with the single point. They imply a probability as shown in Figure 1-1—the only possible outcome is the single point given.

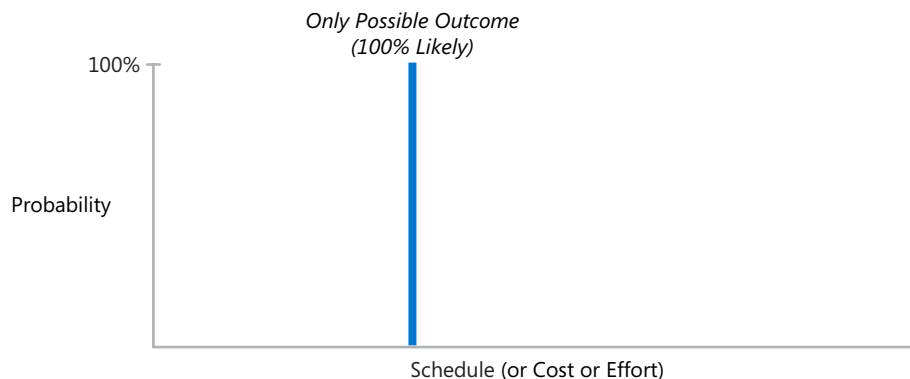


Figure 1-1 Single-point estimates assume 100% probability of the actual outcome equaling the planned outcome. This isn't realistic.

A single-point estimate is usually a target masquerading as an estimate. Occasionally, it is the sign of a more sophisticated estimate that has been stripped of meaningful probability information somewhere along the way.

Tip #3

When you see a single-point “estimate,” ask whether the number is an estimate or whether it’s really a target.

Accurate software estimates acknowledge that software projects are assailed by uncertainty from all quarters. Collectively, these various sources of uncertainty mean that project outcomes follow a probability distribution—some outcomes are more likely, some outcomes are less likely, and a cluster of outcomes in the middle of the distribution are most likely. You might expect that the distribution of project outcomes would look like a common bell curve, as shown in Figure 1-2.

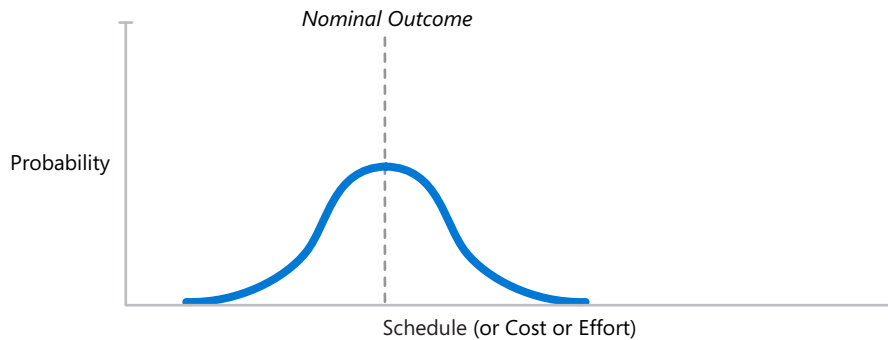


Figure 1-2 A common assumption is that software project outcomes follow a bell curve. This assumption is incorrect because there are limits to how efficiently a project team can complete any given amount of work.

Each point on the curve represents the chance of the project finishing exactly on that date (or costing exactly that much). The area under the curve adds up to 100%. This sort of probability distribution acknowledges the possibility of a broad range of outcomes. But the assumption that the outcomes are symmetrically distributed about the mean (average) is not valid. There is a limit to how well a project can be conducted, which means that the tail on the left side of the distribution is truncated rather than extending as far to the left as it does in the bell curve. And while there is a limit to how well a project can go, there is no limit to how poorly a project can go, and so the probability distribution does have a very long tail on the right.

Figure 1-3 provides an accurate representation of the probability distribution of a software project’s outcomes.

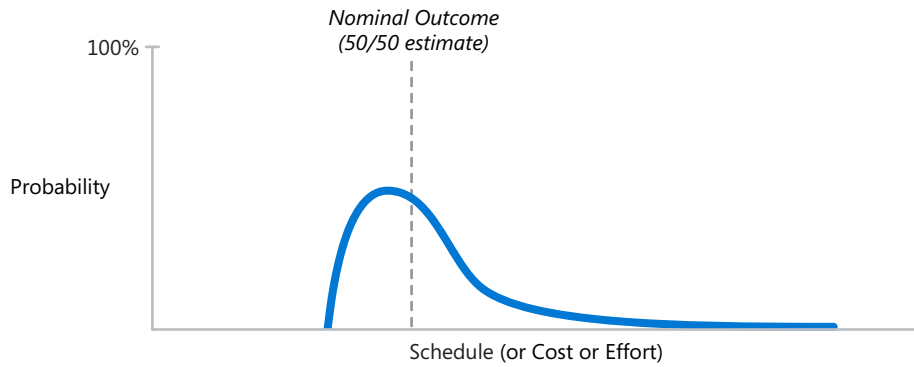


Figure 1-3 An accurate depiction of possible software project outcomes. There is a limit to how well a project can go but no limit to how many problems can occur.

The vertical dashed line shows the “nominal” outcome, which is also the “50/50” outcome—there’s a 50% chance that the project will finish better and a 50% chance that it will finish worse. Statistically, this is known as the “median” outcome.

Figure 1-4 shows another way of expressing this probability distribution. While Figure 1-3 showed the probabilities of delivering on specific dates, Figure 1-5 shows the probabilities of delivering on each specific date *or earlier*.

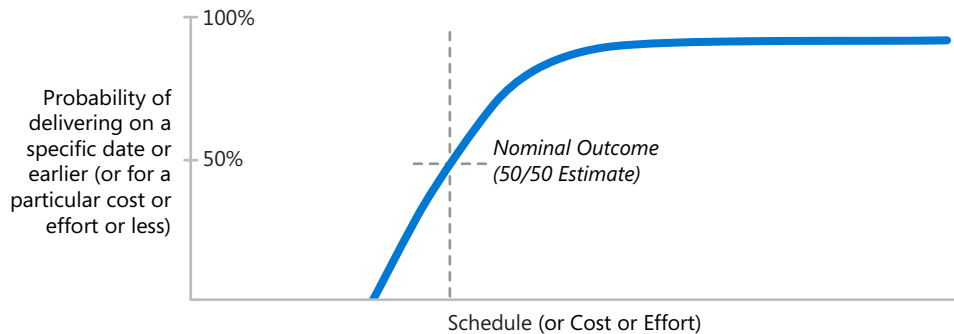


Figure 1-4 The probability of a software project delivering on or before a particular date (or less than or equal to a specific cost or level of effort).

Figure 1-5 presents the idea of probabilistic project outcomes in another way. As you can see from the figure, a naked estimate like “18 weeks” leaves out the interesting information that 18 weeks is only 10% likely. An estimate like “18 to 24 weeks” is more informative and conveys useful information about the likely range of project outcomes.

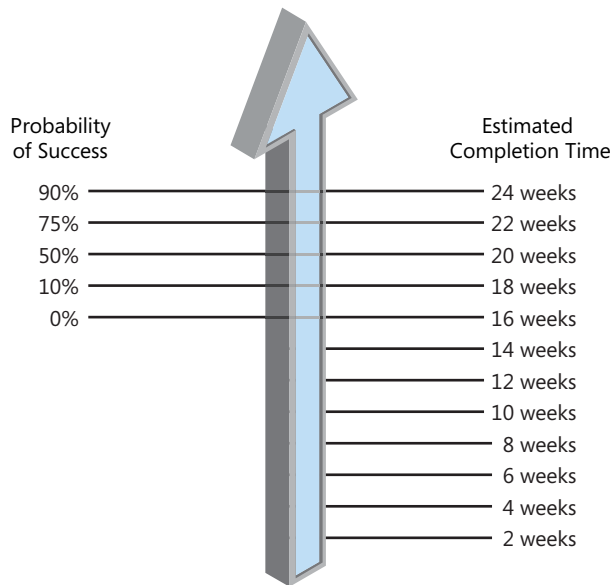


Figure 1-5 All single-point estimates are associated with a probability, explicitly or implicitly.

Tip #4

When you see a single-point estimate, that number's probability is not 100%. Ask what the probability of that number is.

You can express probabilities associated with estimates in numerous ways. You could use a “percent confident” attached to a single-point number: “We’re 90% confident in the 24-week schedule.” You could describe estimates as best case and worst case, which implies a probability: “We estimate a best case of 18 weeks and a worst case of 24 weeks.” Or you could simply state the estimated outcome as a range rather than a single-point number: “We’re estimating 18 to 24 weeks.” The key point is that all estimates include a probability, whether the probability is stated or implied. An explicitly stated probability is one sign of a good estimate.

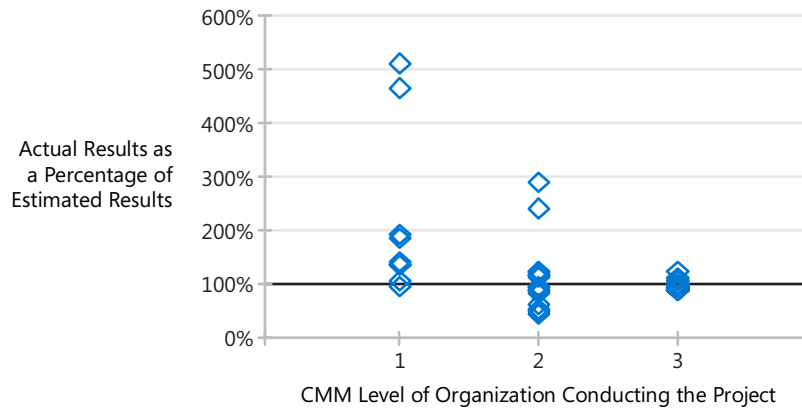
You can make a commitment to the optimistic end or the pessimistic end of an estimation range—or anywhere in the middle. The important thing is for you to know where in the range your commitment falls so that you can plan accordingly.

1.5 Common Definitions of a “Good” Estimate

The answer to the question of what an “estimate” is still leaves us with the question of what a *good* estimate is. Estimation experts have proposed various definitions of a good estimate. Capers Jones has stated that accuracy with $\pm 10\%$ is possible, but only on well-controlled projects (Jones 1998). Chaotic projects have too much variability to achieve that level of accuracy.

In 1986, Professors S.D. Conte, H.E. Dunsmore, and V.Y. Shen proposed that a good estimation approach should provide estimates that are within 25% of the actual results 75% of the time (Conte, Dunsmore, and Shen 1986). This evaluation standard is the most common standard used to evaluate estimation accuracy (Stutzke 2005).

Numerous companies have reported estimation results that are close to the accuracy Conte, Dunsmore, and Shen and Jones have suggested. Figure 1-6 shows actual results compared to estimates from a set of U.S. Air Force projects.



Source: "A Correlational Study of the CMM and Software Development Performance" (Lawlis, Flowe, and Thordahl 1995).

Figure 1-6 Improvement in estimation of a set of U.S. Air Force projects. The predictability of the projects improved dramatically as the organizations moved toward higher CMM levels.¹

Figure 1-7 shows results of a similar improvement program at the Boeing Company.

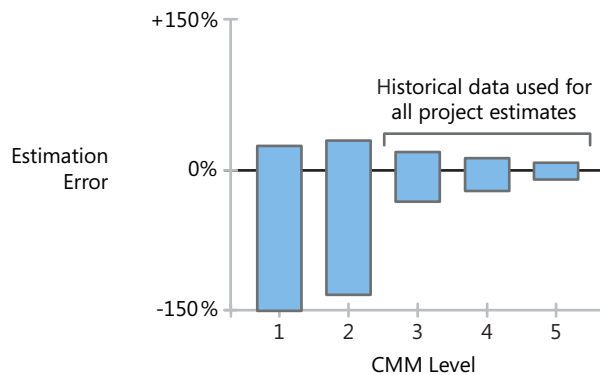


Figure 1-7 Improvement in estimation at the Boeing Company. As with the U.S. Air Force projects, the predictability of the projects improved dramatically at higher CMM levels.

¹ The CMM (Capability Maturity Model) is a system defined by the Software Engineering Institute to assess the effectiveness of software organizations.

A final, similar example, shown in Figure 1-8, comes from improved estimation results at Schlumberger.

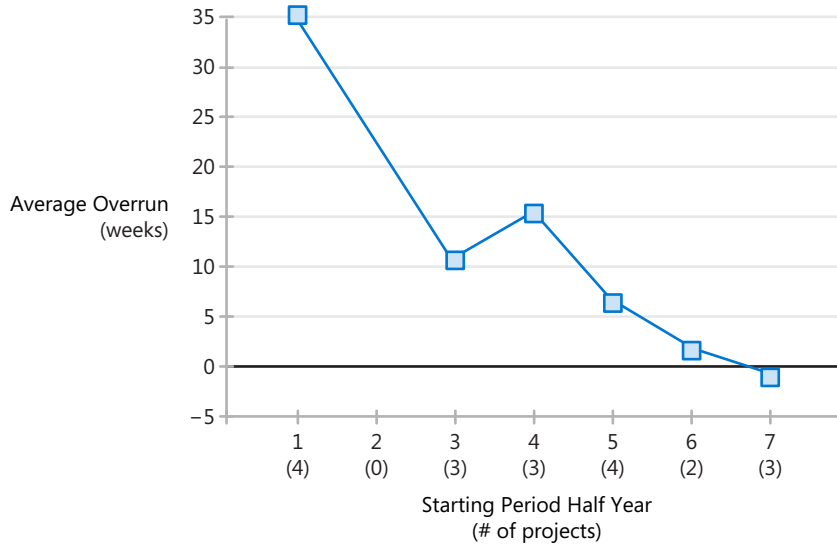


Figure 1-8 Schlumberger improved its estimation accuracy from an average overrun of 35 weeks to an average underrun of 1 week.

One of my client companies delivers 97% of its projects on time and within budget. Telcordia reported that it delivers 98% of its projects on time and within budget (Pitterman 2000). Numerous other companies have published similar results (Putnam and Myers 2003). Organizations are creating good estimates by both Jones’s definition and Conte, Dunsmore, and Shen’s definition. However, an important concept is missing from both of these definitions—namely, that accurate estimation results cannot be accomplished through estimation practices alone. They must also be supported by effective project control.

1.6 Estimates and Project Control

Sometimes when people discuss software estimation they treat estimation as a purely predictive activity. They act as though the estimate is made by an impartial estimator, sitting somewhere in outer space, disconnected from project planning and prioritization activities.

In reality, there is little that is pure about software estimation. If you ever wanted an example of Heisenberg’s Uncertainty Principle applied to software, estimation would be it. (Heisenberg’s Uncertainty Principle is the idea that the mere act of observing a thing changes it, so you can never be sure how that thing would behave if you weren’t observing it.) Once we make an estimate and, on the basis of that estimate, make a

commitment to deliver functionality and quality by a particular date, then we *control* the project to meet the target. Typical project control activities include removing non-critical requirements, redefining requirements, replacing less-experienced staff with more-experienced staff, and so on. Figure 1-9 illustrates these dynamics.

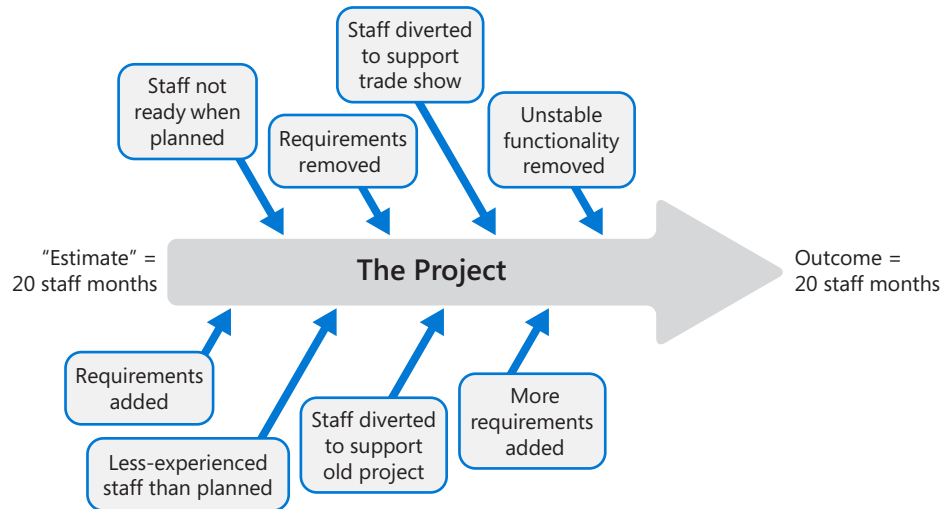


Figure 1-9 Projects change significantly from inception to delivery. Changes are usually significant enough that the project delivered is not the same as the project that was estimated. Nonetheless, if the outcome is similar to the estimate, we say the project met its estimate.

In addition to project control activities, projects are often affected by unforeseen external events. The project team might need to create an interim release to support a key customer. Staff might be diverted to support an old project, and so on.

Events that happen during the project nearly always invalidate the assumptions that were used to estimate the project in the first place. Functionality assumptions change, staffing assumptions change, and priorities change. It becomes impossible to make a clean analytical assessment of whether the project was estimated accurately—because the software project that was ultimately delivered is not the project that was originally estimated.

In practice, if we deliver a project with about the level of functionality intended, using about the level of resources planned, in about the time frame targeted, then we typically say that the project “met its estimates,” despite all the analytical impurities implicit in that statement.

Thus, the criteria for a “good” estimate cannot be based on its predictive capability, which is impossible to assess, but on the estimate’s ability to support project success, which brings us to the next topic: the Proper Role of Estimation.

1.7 Estimation’s Real Purpose

Suppose you’re preparing for a trip and deciding which suitcase to take. You have a small suitcase that you like because it’s easy to carry and will fit into an airplane’s overhead storage bin. You also have a large suitcase, which you don’t like because you’ll have to check it in and then wait for it at baggage claim, lengthening your trip. You lay your clothes beside the small suitcase, and it appears that they will almost fit. What do you do? You might try packing them very carefully, not wasting any space, and hoping they all fit. If that approach doesn’t work, you might try stuffing them into the suitcase with brute force, sitting on the top and trying to squeeze the latches closed. If that still doesn’t work, you’re faced with a choice: leave a few clothes at home or take the larger suitcase.

Software projects face a similar dilemma. Project planners often find a gap between a project’s business targets and its estimated schedule and cost. If the gap is small, the planner might be able to control the project to a successful conclusion by preparing extra carefully or by squeezing the project’s schedule, budget, or feature set. If the gap is large, the project’s targets must be reconsidered.

The primary purpose of software estimation is not to predict a project’s outcome; it is to determine whether a project’s targets are realistic enough to allow the project to be controlled to meet them. Will the clothes you want to take on your trip fit into the small suitcase or will you be forced to take the large suitcase? Can you take the small suitcase if you make minor adjustments? Executives want the same kinds of answers. They often don’t want an accurate estimate that tells them that the desired clothes won’t fit into the suitcase; they want a plan for making as many of the clothes fit as possible.

Problems arise when the gap between the business targets and the schedule and effort needed to achieve those targets becomes too large. I have found that if the initial target and initial estimate are within about 20% of each other, the project manager will have enough maneuvering room to control the feature set, schedule, team size, and other parameters to meet the project’s business goals; other experts concur (Boehm 1981, Stutzke 2005). If the gap between the target and what is actually needed is too large, the manager will not be able to control the project to a successful conclusion by making minor adjustments to project parameters. No amount of careful packing or sitting on the suitcase will squeeze all your clothes into the smaller suitcase, and you’ll have to take the larger one, even if it isn’t your first choice, or you’ll have to leave some clothes behind. The project targets will need to be brought into better alignment with reality before the manager can control the project to meet its targets.

Estimates don’t need to be perfectly accurate as much as they need to be *useful*. When we have the combination of accurate estimates, good target setting, and good planning and control, we can end up with project results that are close to the

“estimates.” (As you’ve guessed, the word “estimate” is in quotation marks because the project that was estimated is not the same project that was ultimately delivered.)

These dynamics of changing project assumptions are a major reason that this book focuses more on the art of estimation than on the science. Accuracy of $\pm 5\%$ won’t do you much good if the project’s underlying assumptions change by 100%.

1.8 A Working Definition of a “Good Estimate”

With the background provided in the past few sections, we’re now ready to answer the question of what qualifies as a good estimate.

A good estimate is an estimate that provides a clear enough view of the project reality to allow the project leadership to make good decisions about how to control the project to hit its targets..

This definition is the foundation of the estimation discussion throughout the rest of this book.

Additional Resources

Conte, S. D., H. E. Dunsmore, and V. Y. Shen. *Software Engineering Metrics and Models*. Menlo Park, CA: Benjamin/Cummings, 1986. Conte, Dunsmore, and Shen’s book contains the definitive discussion of evaluating estimation models. It discusses the “within 25% of actual 75% of the time” criteria, as well as many other evaluation criteria.

DeMarco, Tom. *Controlling Software Projects*. New York, NY: Yourdon Press, 1982. DeMarco discusses the probabilistic nature of software projects.

Stutzke, Richard D. *Estimating Software-Intensive Systems*. Upper Saddle River, NJ: Addison-Wesley, 2005. Appendix C of Stutzke’s book contains a summary of measures of estimation accuracy.

Index

Numeric

90% confidence, 119

A

accuracy of estimates. *See also* accuracy of estimation method; sources of estimation error
benefits of, 27–29
counting error. *See* counting
expressing uncertainty, 251–255
flow of estimates (reestimating), 171–180
chronological, 173–175
poorly estimated projects, 171
recalibrating after missed milestones, 175–179
scheduling reestimation, 177–178
well-estimated projects, 172–173
historical data and, 91–95
need for, 13
overestimation vs. underestimation, 21–24
precision vs., 51–52
reviewing. *See* reviews of estimates
software industry track record, 24–27
accuracy of estimation method. *See also* specific estimation technique by name
choice of technique and, 80
counting error, 85, 88. *See also* counting
effort estimates, 218
expert judgment, 88, 152
expressing uncertainty, 251–255
problems with common techniques, 30
project size estimates, 206
schedule estimates, 231
activities. *See* tasks and responsibilities
activity-based work breakdown structure. *See* WBS
adjusted function-point totals, 201–202
adjustment factors, Cocomo II, 67–70
administrative support, effort of, 247
aggregate best and worst cases, 120–126
allocating effort to various tasks, 233, 237
analogy, estimation by, 127–133
basic approach, 127–132
effort estimates, 209–210
obtaining detailed information, 128

project size, estimating, 205
schedule, estimated, 223–224
size comparisons, 129
uncertainty with, 128
Angel (Analogy Software Tool), 163
applicability tables, explained, 81
Approved Product Definition phase, 35, 39. *See also* Cone of Uncertainty
iterative development, 40–41
unstable project requirements, 42, 247
requirements omitted from estimates, 44–46, 110
software to account for, 160
assessing estimates. *See* reviews of estimates
assumptions in estimation, communicating, 249–251
authority, estimation software as, 160–162
average data for industry, 91, 100–102
defect production, 242
effort graphs, 210–216
ISBSG method, 216–218
average outcome, 8, 59. *See also* diseconomies of scale
averages, meaningful, 86
avoiding bias. *See* bias and subjectivity

B

Basic Schedule Equation, 221–223
Basic Schedule Equation, the (Equation #17), 221
best-case scenarios, 9, 37–39, 107–108. *See also* ranges of estimation
adding to worst case, 118–120
for multiple tasks, creating, 120–126
presenting, 254
shortest possible schedules, 226–228
bias and subjectivity
avoiding with historical data, 93
with estimation software, 162
expert judgment, 89
organizational influences, 92
politics. *See* politics
bottom-up estimation. *See* decomposition
budget accuracy, 28. *See also* costs
budget-driven scheduling, 261
budget estimate (stage-gate processes), 184, 185

- buffers for risk contingency, 245–247
 - quantifying risk, 251–252
- build support, effort of, 247
- burdened costs, 241
- by engineering procedure. *See* decomposition

C

- calendar schedules
 - allocating to various activities, 238–239
 - budget-driven, 261
 - buffers for risk contingency, 245–247
 - quantifying risk, 251–252
 - comparing to past projects, 223–224
 - data on calendar time measures, 95
 - estimating, 221–232
 - basic equation for, 221–223
 - comparing estimates, 231–232
 - Jones's first-order estimation, 224–225
 - reestimation, 177–178
 - with software, 225
 - ownership of, 268
 - predictability of, 29
 - presenting
 - coarse dates and time periods, 255
 - confidence factors, 252–254
 - plus-or-minus qualifiers, 251
 - recalibrating after missed milestones, 175–179
 - shortest possible schedules, 226–228
 - staffing constraints and, 230
 - tradeoffs with effort, 227, 228–230
- calendar time measures, data on, 97
- calibration, 91, 98–102
 - of estimation software, 158, 162
 - with historical data, 98–99
 - with industry average data, 100–102
 - with project data, 99
 - recalibrating after missed milestones, 175–179
- case-based estimates, 254
- changed requirements. *See* unstable project requirements
- Chaos Report (Standish Group), 24
- chaotic development processes, 41
- checklists of project tasks, 110
- chronological estimation flow, 173–175
- clerical support, effort of, 247
- coarse dates, expressing, 255
- Cocomo II tool, 47, 65–70, 163
 - adjustment factors, 67–70
 - effort in estimation, 86
 - effort multipliers, 66
 - personnel considerations, 63
 - scaling factors, 70
- collaboration. *See* communication
- collecting data. *See* data
- Collusion of Optimists, 47
- commitments, 4–6, 173, 269
 - case-based estimates, 254
 - changing, 178
 - Cone of Uncertainty and, 40
 - negotiating, 261. *See also* principled negotiation
 - ranges of estimation and, 257
- common estimation techniques, problems with, 30
- communication
 - about targets, 4–6
 - feedback loops. *See* reviews of estimates
 - presenting estimates, 249–257
 - assumption communication, 249–251
 - expressing uncertainty, 251–255
 - ranges, 256–257
 - reestimation needs, 176–179
 - stage-gate processes, 184
 - unrealistic project expectations, 160
- companies, multiple, 247
- comparing estimates to actuals, 110–112
- completion progress
 - likelihood of completing on time, 22
 - requirements changes and, 42
 - visibility of completion status, 27
- complex standard deviation formula, 122–124
- Complex Standard Deviation Formula (Equation #5), 122
- compressing schedules, 226–228, 229
- computation, 83–84
 - converting counts to estimates, 86–88
 - stage-gate processes, 187
- Cone of Uncertainty, 35–41
 - reestimating throughout project, 173–175
 - schedule estimates and, 222
- confidence, 9
 - 90% confidence, 16–17
 - estimating confidence factors, 252
- configuration management, effort of, 247
- Construx Estimate tool, 60, 163
 - computing effort with, 210
 - computing schedule with, 225
- contingency buffers, 245–247, 251–252
- control. *See* project control

convergence between estimation approaches. *See*
 multiple estimation approaches
 correcting errors in software, 241–245
 Costar tool, 163
 costs
 budget accuracy, 28
 budget-driven scheduling, 261
 estimating, 241
 predictability of, 29
 range presentation and, 256
 counting, 84–86
 computing estimates from, 86–88
 function points, 200
 stage-gate processes, 187
 credibility of development team, 28
 creeping requirements, 42, 247
 requirements omitted from estimates, 44–46,
 110
 software to account for, 160

D

data
 calibrating with. *See* calibration
 comparing estimates to actuals, 110–112
 counted, 84–86
 computing estimates from, 86–88
 function points, 200
 stage-gate processes, 187
 estimation software requirements, 162
 historical, 91–98
 accuracy with, 91–95
 averages of, determining, 136
 collecting, effort of, 235
 comparing to past projects. *See* analogy,
 estimation by
 estimation software with, 158
 expert judgment vs., 89
 what to collect, 95–98
 industry averages, 91, 100–102
 defect production, 242
 effort graphs, 210–216
 ISBSG method, 216–218
 project data, 91, 99
 dates. *See* schedules (calendar)
 debating estimates, 172
 debugging considerations, 241–245
 decomposition, 113–126
 estimating project size with, 205
 granularity of estimated tasks, 116
 work breakdown structures for. *See* WBS

defect measures, data on, 95, 97
 defect production and removal, 241–245
 defining product. *See* Product Definition phase
 delivery
 confidence factors, 252–254
 evolutionary delivery, 79
 promises of. *See* commitments
 staged, 79
 targets, 3, 173, 269
 communicating about, 4–6
 determining if realistic, 13
 Delphi technique, 150–155
 density of errors, project size and, 242
 Detailed Design Complete phase, 35, 39. *See also*
 Cone of Uncertainty
 developer-to-tester ratios, 237
 development effort, 207–219
 with fuzzy logic, 138
 with industry average data, 210–216
 influences on, 207–208
 from project size, 209–210
 software for, 158
 development language, influence on estimates,
 64–65, 247
 development process
 controlling. *See* project control
 stage-gate processes, 182–185
 stages of, 80
 counting, 85
 development style, 78–79
 direct costs, 241
 discussing estimates. *See* communication;
 principled negotiation
 diseconomies of scale, 56–61, 70
 effort estimates, 208
 modeling after historical data, 99
 software to account for, 160
 when unimportant, 60
 documenting estimation processes, 182
 assumptions, 249–251
 doubt, expressing, 251–255
 Dutch method of function-point estimation, 203,
 205
 Dutch Method's Indicative Function Point Count
 Formula (Equation #8), 203

E

early development stage, 80
 counting at, 85
 education of nontechnical staff, 263

- efficiency of defect removal, 242–245
- effort multipliers, Cocomo II model, 66
- effort, project
 - allocating to various activities, 233–238
 - collecting data on, 95, 96
 - comparing estimates for, 218
 - computing, 210
 - estimating, 207–219
 - with fuzzy logic, 138
 - with industry average data, 210–216
 - influences on, 207–208
 - from project size, 209–210
 - software for, 158
 - practical considerations (ideal vs. planned), 239–241
 - tradeoffs with scheduling, 227, 228–230
- EMs (effort multipliers), Cocomo II model, 67–70
- environment, programming, 65
- equations
 - Basic Schedule Equation, the, 221
 - Complex Standard Deviation Formula, 122
 - Dutch Method's Indicative Function Point Count Formula, 203
 - Informal Comparison to Past Projects Formula, 223
 - ISBSG Effort Formula for Desktop Projects, 217
 - ISBSG Effort Formula for Enhancement Projects, 217
 - ISBSG Effort Formula for Fourth Generation Projects, 217
 - ISBSG Effort Formula for General Projects, 216
 - ISBSG Effort Formula for Mainframe Projects, 216
 - ISBSG Effort Formula for Mid-Range Projects, 217
 - ISBSG Effort Formula for New Development Projects, 217
 - ISBSG Effort Formula for Third Generation Projects, 217
 - Magnitude of Relative Error (MRE) Formula, 110
 - Modified Complex Standard Deviation Formula, 124
 - PERT Formula for Estimating Number of Components, 139
 - Pessimistic PERT Formula, 109
 - Program Evaluation and Review Technique (PERT) Formula, 109
 - Simple Standard Deviation Formula, 121
- errors in project. *See* software quality
- errors of estimation. *See* sources of estimation error
- estimate accuracy. *See also* sources of estimation error
 - benefits of, 27–29
 - counting error. *See* counting
 - expressing uncertainty, 251–255
 - flow of estimates (reestimating), 171–180
 - chronological, 173–175
 - poorly estimated projects, 171
 - recalibrating after missed milestones, 175–179
 - scheduling reestimation, 177–178
 - well-estimated projects, 172–173
 - historical data and, 91–95
 - need for, 13
 - overestimation vs. underestimation, 21–24
 - precision vs., 51–52
 - reviewing. *See* reviews of estimates
 - software industry track record, 24–27
- Estimate Express tool, 163
- estimate influences, 55–72
 - diseconomies of scale, 56–61, 70
 - effort estimates, 208
 - modeling after historical data, 99
 - software to account for, 160
 - when unimportant, 60
- kind of software, 61–63, 236
- miscellaneous other sources, 65–70
- personnel factors, 63
- political, 260–263
- programming language, 64–65
- size. *See* project size
- estimate precision, unwarranted, 51–52
- estimate presentation styles, 249–257
 - assumption communication, 249–251
 - expressing uncertainty, 251–255
 - ranges, 256–257
- estimate rejection, 262
- estimates, communicating about, 4–6
- estimates, debating, 172, 269. *See also* negotiation and problem solving
- estimates, defined, 3–14, 173
 - common definitions, 9
 - good estimates, 9
 - plans vs., 4
 - probability statements, 6–9
 - working definition, 14
- estimates, reviewing. *See* reviews of estimates
- estimating models, industry differences in, 63
- estimating software size. *See* size, project

- estimation error, sources of, 33–53. *See also*
 - accuracy of estimates; accuracy of estimation method
- chaotic development processes, 41
- Cone of Uncertainty, 35–41
 - reestimating throughout project, 173–175
 - schedule estimates and, 222
- miscellaneous other sources, 52
- off-the-cuff estimates, 49–51
- omitted activities, 44–46
- politics, 259–270
 - attributes of executives, 259–260
 - avoiding with historical data, 93–95
 - influences on estimates, 260–263
 - negotiation and problem solving, 259–260, 261, 263–270
- subjectivity and bias, 47–49
- unfounded optimism, 46
- unstable project requirements, 42, 247
 - requirements omitted from estimates, 44–46, 110
 - software to account for, 160
 - unwarranted precision, 51–52
- estimation flow (reestimating), 171–180
 - chronological, 173–175
 - poorly estimated projects, 171
 - recalibrating after missed milestones, 175–179
 - scheduling reestimation, 177–178
 - well-estimated projects, 172–173
- estimation negotiation, 263–270
 - attributes of executives and, 259–260
 - estimates vs. commitments, 261
- estimation politics, 259–270
 - attributes of executives and, 259–260
 - avoiding with historical data, 93–95
 - influences on estimates, 260–263
 - negotiation and problem solving, 263–270
 - attributes of executives and, 259–260
 - estimates vs. commitments, 261
- estimation skill, testing, 15–19, 273
- estimation software, 157–164
 - calibrating, 162
 - computing effort with, 210
 - computing schedule with, 225
 - estimating project size with, 205
 - list of, 163–164
- estimation techniques, 77–81, 80
 - by analogy, 127–133
 - basic approach, 127–132
 - effort estimates, 209–210
 - estimating project size, 205
 - estimating schedule, 223–224
 - obtaining detailed information, 128
 - size comparisons, 129
 - uncertainty with, 128
 - applicability tables, explained, 81
 - computation, 83–84
 - converting counts to estimates, 86–88
 - stage-gate processes, 187
 - counting, 84–86
 - computing estimates with, 86–88
 - function points, 200
 - stage-gate processes, 187
 - Delphi technique, 150–155
 - how to choose, 77–80
 - judgment. *See* expert judgment; off-the-cuff estimates
 - multiple approaches, using, 165–169
 - for effort, 218
 - for project size, 206
 - ranges, presenting, 256
 - for schedules, 231–232
 - stage-gate processes, 187
 - proxy-based, 135–147
 - fuzzy logic, 136–138, 205
 - standard components approach, 138–141
 - story points, 142–144, 205
 - t-shirt sizing, 145–146
- events, unforeseen, 12
- evolutionary delivery, 79
- evolutionary prototyping, 79
- executives, attributes of, 259–260
- expected-case calculation, 108–109, 120
 - complex standard deviation formula, 122–124
 - by decomposition. *See* decomposition
 - simple standard deviation formula, 121–122
- expert judgment, 83–84, 88–89, 105–112
 - comparing to actuals, 110–106
 - group (structured) judgment, 106–110, 149–155
 - estimating project size with, 205
 - Wideband Delphi technique, 150–155
 - insisting on objective criteria, 268
- exploratory estimate (stage-gate processes), 184, 185, 188
- exponential increases with project size. *See* diseconomies of scale
- extending schedule to reduce effort, 228
- external bias, 48
- external I/O, as function points, 200

external interface files, as function points, 200
 external political constraints, 260
 extreme programming, 79
 story points, 142–144, 205

F

fantasy factor. *See* optimism, unfounded
 features. *See* functionality, program
 feedback loop for estimations review. *See* reviews
 of estimates
 Fibonacci sequences, 143
 final commitment estimate (stage-gate processes),
 184, 186
 First-Order Estimation Practice, 224–225
 first-time-development effort factors, 247
 flow of estimates (reestimating), 171–180
 chronological, 173–175
 poorly estimated projects, 171
 recalibrating after missed milestones, 175–179
 scheduling reestimation, 177–178, 187
 well-estimated projects, 172–173
 forgetting activities when estimating, 44–46, 110
 formula for Expected Case, 108–109
 formulas
 Basic Schedule Equation, the, 221
 Complex Standard Deviation, 122
 Dutch Method's Indicative Function Point
 Count, 203
 Informal Comparison to Past Projects, 223
 ISBSG Effort Formula for Desktop Projects, 217
 ISBSG Effort Formula for Enhancement
 Projects, 217
 ISBSG Effort Formula for Fourth Generation
 Projects, 217
 ISBSG Effort Formula for General Projects, 216
 ISBSG Effort Formula for Mainframe Projects,
 216
 ISBSG Effort Formula for Mid-Range Projects,
 217
 ISBSG Effort Formula for New Development
 Projects, 217
 ISBSG Effort Formula for Third Generation
 Projects, 217
 Magnitude of Relative Error (MRE), 110
 Modified Complex Standard Deviation, 124
 PERT Formula for Estimating Number of
 Components, 139
 Pessimistic PERT, 109

Program Evaluation and Review Technique
 (PERT), 109
 Simple Standard Deviation, 121
 functionality, program
 case-based estimates, 254, 255
 classifying, 137
 defect production and, 242
 function-point estimation, 200–205
 converting to LOC, 202–203
 simplified techniques, 203–205
 predictability of, 29
 function-point estimation, 200–205
 fuzzy logic, 136–138, 205

G

Goldratt's Student Syndrome, 22, 23
 good estimate, defined, 9
 granularity of estimated tasks, 106, 116
 group expert judgment, 106–110, 149–155
 estimating project size with, 205
 Wideband Delphi technique, 150–155
 group reviews of estimates, 111, 149–150. *See also*
 structured expert judgment
 with estimation software, 162
 of project size, 205
 sanity checking, 162, 271
 with standardized estimation, 192
 growth in requirements, 43
 guess-based estimation. *See* off-the-cuff estimates
 GUI elements, counting, 204, 205

H

historical data, 91–98
 accuracy with, 91–95
 averages of, determining, 136
 calibrating with. *See* calibration
 collecting, effort of, 235
 comparing to past projects. *See* analogy,
 estimation by
 estimation software with, 158
 expert judgment vs., 89
 what to collect, 95–98
 human resource considerations, 63

I

ideal effort vs. planned effort, 239–241
 impartial, estimation software as, 160–162
 Impossible Zone, 227

- inaccuracy. *See* accuracy of estimates; accuracy of estimation method
 - individual expert judgment, 83–84, 88–89, 105–112. *See also* group expert judgment
 - comparing to actuals, 110–106
 - insisting on objective criteria, 268
 - industry average data, 91, 100–102
 - defect production, 242
 - effort graphs, 210–216
 - ISBSG method, 216–218
 - industry differences in productivity, 62
 - inequalities of scale, 56–61, 70
 - effort estimates, 208
 - modeling after historical data, 99
 - software to account for, 160
 - when unimportant, 60
 - influences on estimates, 55–72
 - diseconomies of scale, 56–61, 70
 - effort estimates, 208
 - modeling after historical data, 99
 - software to account for, 160
 - when unimportant, 60
 - kind of software, 61–63, 236
 - miscellaneous other sources, 65–70
 - personnel factors, 63
 - political, 260–263
 - programming language, 64–65
 - size. *See* project size
 - influences on project effort, 207–208
 - informal analogy-based estimation. *See* off-the-cuff estimates
 - Informal Comparison to Past Projects Formula (Equation #18), 223
 - informal comparisons. *See* analogy, estimation by
 - informal estimation. *See also* expert judgment
 - Initial Concept phase. *See also* Cone of Uncertainty
 - commitment at, 40
 - estimation error from, 39
 - interface files, as function points, 200
 - interim releases, 12, 23
 - internal bias. *See* subjectivity and bias
 - internal logical files, as function points, 200
 - international effort factors, 247
 - interpreted programming languages, 65
 - intuition-based estimation. *See* off-the-cuff estimates
 - ISBSG Effort Formula for Desktop Projects (Equation #12), 217
 - ISBSG Effort Formula for Enhancement Projects (Equation #15), 217
 - ISBSG Effort Formula for Fourth Generation Projects (Equation #14), 217
 - ISBSG Effort Formula for General Projects (Equation #9), 216
 - ISBSG Effort Formula for Mainframe Projects (Equation #10), 216
 - ISBSG Effort Formula for Mid-Range Projects (Equation #11), 217
 - ISBSG Effort Formula for New Development Projects (Equation #16), 217
 - ISBSG Effort Formula for Third Generation Projects (Equation #13), 217
 - ISBSG method for computing effort, 216–218
 - IT support, effort of, 247
 - iterative development, 40–41, 78, 79
 - stages of development, 80
 - standardized estimation procedure for, 188–189
- J**
- Jones's first-order estimation, 224–225
 - judgment, using to estimate, 83–84, 88–89, 105–112
 - comparing to actuals, 110–106
 - group (structured) judgment, 106–110, 149–155
 - estimating project size, 205
 - Wideband Delphi technique, 150–155
 - insisting on objective criteria, 268
- K**
- KLOC. *See* lines of code (LOC)
 - KnowledgePLAN tool, 163
- L**
- language, influence on estimates, 64–65, 247
 - large numbers, law of, 115, 137
 - large projects, 78. *See also* project size
 - error density and, 242
 - estimation flow (reestimating), 174
 - late development stage, 80
 - counting at, 85
 - schedule estimates, 223
 - late projects
 - how late (software industry), 26
 - recalibrating after missed milestones, 175–179
 - unnecessary activities from, 23

Law of Large Numbers, 115, 137
 lengthening schedule to reduce effort, 228
 lines of code (LOC), 55–56
 collecting data on, 95
 converting function points to, 202–203
 size estimation with, 198–200
 logical files, as function points, 200

M

magnitude of relative error (MRE), 110
 Magnitude of Relative Error (MRE) Formula
 (Equation #3), 110
 management effort, 235
 management (executive), attributes of, 259–260
 median outcome, 8, 59. *See also* diseconomies of scale
 medium-size projects, 78. *See also* project size
 memory-based estimation. *See* off-the-cuff estimates
 methods of estimation, 77–81
 by analogy, 127–133
 basic approach, 127–132
 effort estimates, 209–210
 estimating project size with, 205
 estimating schedule, 223–224
 obtaining detailed information, 128
 size comparisons, 129
 uncertainty with, 128
 applicability tables, explained, 81
 computation, 83–84
 converting counts to estimates, 86–88
 stage-gate processes, 187
 counting, 84–86
 computing estimates with, 86–88
 function points, 200
 stage-gate processes, 187
 Delphi technique, 150–155
 how to choose, 77–80
 judgment. *See* expert judgment; off-the-cuff estimates
 multiple approaches, using, 165–169
 for effort, 218
 for project size, 206
 ranges, presenting, 256
 for schedules, 231–232
 stage-gate processes, 187
 proxy-based, 135–147
 fuzzy logic, 136–138, 205
 standard components approach, 138–141

 story points, 142–144, 205
 t-shirt sizing, 145–146
 software. *See* estimation software
 micro-estimation. *See* decomposition
 middle development stage, 80
 counting at, 85
 midpoint of best- and worst-case estimates, 108
 midpoints of ranges, 256
 milestones, missed. *See* flow of estimates
 minus qualifiers, 251
 missed milestones. *See* flow of estimates
 missing requirements, 44–46, 110
 avoiding with work breakdown structures. *See* WBS
 mistakes in estimation. *See* accuracy of estimates; accuracy of estimation method
 Modified Complex Standard Deviation Formula
 (Equation #6), 124
 module buildup. *See* decomposition
 Monte Carlo simulation, 158
 most-likely-case estimate, 108–109, 120
 MRE (magnitude of relative error), 110
 multiple-company effort factors, 247
 multiple estimation approaches, 165–169
 for effort, 218
 for project size, 206
 ranges, presenting, 256
 for schedules, 231–232
 stage-gate processes, 187
 mutual gain, options for, 266

N

narrow ranges of estimation. *See* width of estimation ranges
 negotiation and problem solving, 263–270
 attributes of executives and, 259–260
 estimates vs. commitments, 261
 net business values, 146
 new development, defined, 80
 nominal outcome, 8, 59. *See also* diseconomies of scale
 nominal schedules, compressing, 226–228
 nonsoftware activities
 coordinating with software activities, 28
 omitted from estimates, 45
 nontechnical staff, educating, 263
 numeric scales, cautions about, 143–144

O

objective criteria, insisting on, 268
 objective, estimation software as, 160–162
 off-the-cuff estimates, 31, 49–51. *See also* expert judgment
 comparing to actuals, 110–106
 insisting on objective criteria, 268
 omitting activities when estimating, 44–46
 on-time completion, likelihood of, 22. *See also* completion progress
 opinion of experts. *See* expert judgment
 optimism, unfounded, 46, 119
 organizational influences, accounting for, 92
 outcomes
 Chaos Report (Standish Group), 24
 diseconomies of scale, 56–61, 70
 effort estimates, 208
 modeling after historical data, 99
 software to account for, 160
 when unimportant, 60
 kind of software (industry differences), 61, 236
 nominal, defined, 8
 organizational influences on, 92
 personnel factors, 63
 probabilistic, 6–9
 programming languages and, 64–65
 project size and. *See* project size
 simulating, 157
 overestimation, advantages of, 21–24
 overlooked responsibilities, 44–46, 110
 overtime effort, compensating for, 241
 ownership of schedules, 268

P

Parkinson's Law, 21, 24
 past projects, comparing to. *See* analogy, estimation by
 percent confidence, 9
 complex standard deviation formula, 122–124
 pitfalls with, 126
 simple standard deviation formula, 121–122
 percentiles, standard components with, 140–141
 personal memory, estimating from. *See* off-the-cuff estimates
 personnel
 attributes of executives, 259–260
 education of nontechnical staff, 263
 influence on estimates, 63

 reducing estimation politics, 93–95
 staffing constraints and schedule estimates, 230. *See also* project effort
 PERT Formula for Estimating Number of Components (Equation #7), 139
 PERT (Program Evaluation and Review Technique), 108–109
 Pessimistic PERT Formula (Equation #2), 109
 planning
 breaking negotiation deadlocks, 267
 estimating vs., 4
 options for, calculating, 160
 parameters for, 233–248
 activity breakdown, 233–238
 allocating time for activities, 238–239
 cost estimates, 241
 defect production and removal, 241–245
 effort estimates, making practical, 239–241
 risk and contingency buffers, 245–247, 251–252
 various support needs, 247
 reduced effectiveness from underestimation, 22
 reestimation, 177–178
 visibility of completion status, 27
 what-if analysis, 160
 plus-or-minus qualifiers, 251
 politics, 259–270
 attributes of executives, 259–260
 avoiding with historical data, 93–95
 influences on estimates, 260–263
 negotiation and problem solving, 263–270
 attributes of executives and, 259–260
 estimates vs. commitments, 261
 poorly estimated projects, 171, 176–177, 179. *See also* single-point estimates
 precision, unwarranted, 51–52
 predictability, as important, 29–30
 preliminary commitment estimate (stage-gate processes), 186
 presenting estimates, 249–257
 assumption communication, 249–251
 expressing uncertainty, 251–255
 ranges, 256–257
 pressure to create narrow ranges, 18
 preventing bias. *See* bias and subjectivity
 Price-S tool, 163
 principled negotiation, 263–270
 attributes of executives and, 259–260
 estimates vs. commitments, 261

- probabilistic project outcomes, 6–9
- probability analysis, 158
- problem solving, 263–270
- process. *See* project control
- producing defects, 241–245
- Product Definition phase, 35. *See also* Cone of Uncertainty
 - commitment at, 40
 - estimation error from, 39
 - unstable project requirements, 42, 247
 - requirements omitted from estimates, 44–46, 110
 - software to account for, 160
- productivity data, 208
- products
 - delivery of
 - confidence factors, 252–254
 - evolutionary delivery, 79
 - promises of. *See* commitments
 - staged, 79
 - targets. *See* targets
 - functionality of
 - case-based estimates, 254, 255
 - classifying, 137
 - defect production and, 242
 - function-point estimation, 200–205
 - predictability of, 29
 - kinds of, 61–63, 236
 - quality of
 - accuracy estimates and, 27
 - defect production and removal, 241–245
- Program Evaluation and Review Technique (PERT) Formula (Equation #1), 109
- program functionality
 - case-based estimates, 254, 255
 - classifying, 137
 - defect production and, 242
 - function-point estimation, 200–205
 - converting to LOC, 202–203
 - simplified techniques, 203–205
 - predictability of, 29
- programming. *See* entries at development
- programming language, influence on estimates, 64–65, 247
- programs. *See* products
- progress tracking
 - requirements changes and, 42
 - visibility of completion status, 27
- project control, 11–14. *See also* tasks and responsibilities
 - activities overlooked in estimates, 44–46
 - chaotic development processes, 41
 - communication. *See* communication
 - reduced effectiveness from underestimation, 22
 - Student Syndrome, addressing, 23
 - unstable project requirements, 42, 247
 - requirements omitted from estimates, 44–46, 110
 - software to account for, 160
 - what-if analysis, 160
- project data, 91, 99
- project effort
 - allocating to various activities, 233–238
 - comparing estimates for, 218
 - computing, 210
 - estimating, 207–219
 - with fuzzy logic, 138
 - with industry average data, 210–216
 - influences on, 207–208
 - from project size, 209–210
 - software for, 158
- practical considerations (ideal vs. planned), 239–241
- tradeoffs with scheduling, 227, 228–230
- project outcomes
 - Chaos Report (Standish Group), 24
 - diseconomies of scale, 56–61, 70
 - effort estimates, 208
 - modeling after historical data, 99
 - software to account for, 160
 - when unimportant, 60
- kind of software (industry differences), 61, 236
- nominal, defined, 8
- organizational influences on, 92
- personnel factors, 63
- probabilistic, 6–9
- programming languages and, 64–65
- project size and. *See* project size
- simulating, 157
- project overruns. *See* late projects
- project planning
 - breaking negotiation deadlocks, 267
 - estimating vs., 4
 - options for, calculating, 160
- parameters for, 233–248
 - activity breakdown, 233–238
 - allocating time for activities, 238–239
 - cost estimates, 241
 - defect production and removal, 241–245
 - effort estimates, making practical, 239–241
 - risk and contingency buffers, 245–247, 251–252
 - various support needs, 247
- reduced effectiveness from underestimation, 22

- reestimation, 177–178
- visibility of completion status, 27
- what-if analysis, 160
- project requirements
 - creating, effort for, 234, 239
 - Requirements Complete phase, 35, 39. *See also*
 - Cone of Uncertainty
 - iterative development, 40–41
 - unstable (creeping), 42, 247
 - requirements omitted from estimates, 44–46, 110
 - software to account for, 160
- project size, 25, 55–61
 - choosing as estimation technique, 78
 - collecting historical data on, 95
 - comparing similar projects, 129
 - counting and, 85
 - diseconomies of scale, 56–61, 70
 - effort estimates, 208
 - modeling after historical data, 99
 - software to account for, 160
 - when unimportant, 60
 - error density and, 242
- estimating, 197–206
 - by function points, 200–205
 - with fuzzy logic, 136–138, 205
 - by LOC. *See* lines of code (LOC)
 - measures for, 197
 - standard components, 138–141
 - summary of techniques for, 205–206
- estimating effort from, 209–210
- estimation flow and (reestimating), 174
- estimation software and, 162
- schedule breakdown, 238
- total effort and, 235
- project tasks and responsibilities
 - allocating effort to, 233–238
 - checklists, 110
 - defect production estimates, 241
 - granularity of, 106, 116
 - overall best/worst case, creating, 120–126
 - overlooked in estimates, 44–46, 110
 - total effort breakdown, 235
- promise to deliver. *See* commitments
- proxy-based techniques, 135–147
 - fuzzy logic, 136–138, 205
 - standard components approach, 138–141
 - story points, 142–144, 205
 - t-shirt sizing, 145–146

- public estimation review, 111, 149–150. *See also*
 - structured expert judgment
 - with estimation software, 162
 - of project size, 205
 - sanity checking, 162, 271
 - with standardized estimation, 192

Q

- quality of estimation. *See* accuracy of estimates
- quality of software
 - accuracy estimates and, 27
 - defect production and removal, 241–245
- quantifying risk, 251–252
- quarterly schedules, 261
- quiz for estimation skills, 15–19, 273

R

- ranges of estimation, 107–108
 - communicating, 256–257
 - confidence intervals, 252–254
 - estimate refinement and, 177
 - probabilistic project outcomes, 6–9
 - width of, 18
- ratio-based estimation approach, 60
- Rational Unified Process (RUP), 79
- RE (risk exposure), 246
- recalibrating after missed milestones, 175–179
- recomposition. *See* decomposition
- reducing bias. *See* bias and subjectivity
- reestimating. *See* flow of estimates
- refining estimates. *See* calibration
- rejection of estimates, 262
- relative error, 110
- remembering old estimates, 51
- removing defects from software, 242–245
- reporting estimates, 249–257
 - assumption communication, 249–251
 - expressing uncertainty, 251–255
 - ranges, 256–257
- Requirements Complete phase, 35, 39. *See also*
 - Cone of Uncertainty
 - iterative development, 40–41
 - unstable project requirements, 42, 247
 - requirements omitted from estimates, 44–46, 110
 - software to account for, 160

- requirements, project
 - creating, effort for, 234, 239
 - unstable (creeping), 42, 247
 - requirements omitted from estimates, 44–46, 110
 - software to account for, 160
- responsibilities, project-related
 - allocating effort to, 233–238
 - checklists, 110
 - defect production estimates, 241
 - granularity of, 106, 116
 - overall best/worst case, creating, 120–126
 - overlooked in estimates, 44–46, 110
- reviews of estimates, 111, 149–150. *See also*
 - structured expert judgment
 - comparing estimates to actuals, 110–112
 - with estimation software, 162
 - of project size, 205
 - sanity checking, 162, 271
 - with standardized estimation, 192
- risk and contingency buffers, 245–247, 251–252
- risk information, obtaining early, 28
- risk quantification, 251–252
- RUP (Rational Unified Process), 79

S

- sanity checking, 271
 - with estimation software, 162
- scale, diseconomies of, 70
- scaling factors, Cocomo II, 70
- schedules (calendar)
 - allocating to various activities, 238–239
 - budget-driven, 261
 - buffers for risk contingency, 245–247
 - quantifying risk, 251–252
 - comparing to past projects, 223–224
 - data on calendar time measures, 95
 - estimating, 221–232
 - basic equation for, 221–223
 - comparing estimates, 231–232
 - Jones's first-order estimation, 224–225
 - reestimation, 177–178
 - with software, 225
 - ownership of, 268
 - predictability of, 29
 - presenting
 - coarse dates and time periods, 255
 - confidence factors, 252–254
 - plus-or-minus qualifiers, 251

- recalibrating after missed milestones, 175–179
 - shortest possible schedules, 226–228
 - staffing constraints and, 230
 - tradeoffs with effort, 227, 228–230
- science of estimation. *See* estimation software
- Scrum development style, 79
- SDLC (software development life cycles). *See*
 - stage-gate processes
- SEER tool, 163
- sequential development, 78, 79
 - stages of development, 80
 - standardized estimation procedure for, 185–187
- shortest possible schedules, 226–228
- similar projects, comparing. *See* analogy,
 - estimation by
- simple standard deviation formula, 121–122
- Simple Standard Deviation Formula (Equation #4), 121
- simplified function-point techniques, 203–205
- simulating project outcomes, 157
- single-point estimates, 6, 107–108, 176–177
- size, project, 25, 55–61
 - choosing as estimation technique, 78
 - collecting historical data on, 95
 - comparing similar projects, 129
 - counting and, 85
 - error density and, 242
 - estimating, 197–206
 - by function points, 200–205
 - with fuzzy logic, 136–138, 205
 - by LOC. *See* lines of code (LOC)
 - measures for, 197
 - standard components, 138–141
 - summary of techniques for, 205–206
 - estimating effort from, 209–210
 - estimation flow and (reestimating), 174
 - estimation software and, 162
 - schedule breakdown, 238
 - total effort and, 235
- size, team. *See* project effort
- skills at estimation, testing, 15–19, 273
- SLIM-Estimate tool, 163
- small projects, 78. *See also* project size
 - estimation flow (reestimating), 175
 - schedule estimates, 223
- software development style, 78–79
- software estimates. *See* entries at estimate
- software for estimation, 157–164
 - calibrating, 162
 - computing effort with, 210

- computing schedule with, 225
- estimating project size with, 205
- list of, 163–164
- software functionality. *See* functionality, program
- software industry track record, 24, 27
- software negotiations, 263–270
 - attributes of executives and, 259–260
 - estimates vs. commitments, 261
- software product. *See* products
- software quality
 - accuracy estimates and, 27
 - defect production and removal, 241–245
- sources of estimation error, 33–53. *See also*
 - accuracy of estimates; accuracy of estimation method
- chaotic development processes, 41
- Cone of Uncertainty, 35–41
 - reestimating throughout project, 173–175
 - schedule estimates and, 222
- miscellaneous other sources, 52
- off-the-cuff estimates, 49–51
- omitted activities, 44–46
- politics, 259–270
 - attributes of executives, 259–260
 - avoiding with historical data, 93–95
 - influences on estimates, 260–263
 - negotiation and problem solving, 259–260, 261, 263–270
- subjectivity and bias, 47–49
- unfounded optimism, 46
- unstable project requirements, 42, 247
 - requirements omitted from estimates, 44–46, 110
 - software to account for, 160
 - unwarranted precision, 51–52
- staffing constraints and schedule estimates, 230. *See also* project effort
- stage-gate processes, 182–185
- staged delivery, 79
- stages of development, 80
 - counting, 85
 - stage-gate processes, 182–185
- standard components approach, 145–146
 - estimating project size with, 205
 - with percentiles, 140–141
- standard deviation, 121–126, 256
 - complex formula for best/worst cases, 122–124
 - simple formula for best/worst cases, 121–122
 - squared (variance), 122

- standardized estimation procedures, 173, 181–193, 269. *See also* well-estimated projects
- documenting assumptions, 249–251
- elements of, 181
- example from advanced organization, 190
- how to improve, 192
- for iterative projects, 188–189
- for sequential projects, 185–187
- stage-gate processes, 182–185
- Standish Group's Chaos Report, 24
- status visibility, 27
- story points, 142–144, 205
- structured expert judgment, 106–110, 149–155. *See also* expert judgment
 - estimating project size with, 205
 - Wideband Delphi technique, 150–155
- Student Syndrome, 22, 23
- style of software development, 78–79
- subjectivity and bias, 47–49
 - avoiding with historical data, 93
 - with estimation software, 162
- expert judgment, 89
- organizational influences, 92
- politics. *See also* politics
- support needs, effort for, 247
- support, programming, 65

T

- t-shirt sizing, 145–146
- talking about estimates. *See* communication; principled negotiation
- targets, 3, 173, 269
 - communicating about, 4–6
 - determining if realistic, 13
- tasks and responsibilities (project-related)
 - allocating effort to, 233–238
 - checklists, 110
 - defect production estimates, 241
 - granularity of, 106, 116
 - overall best/worst case, creating, 120–126
 - overlooked in estimates, 44–46, 110
- team credibility, 28. *See also* communication
- team estimates. *See* structured expert judgment
- team size. *See* project effort
- technical foundation, underestimation and, 22

techniques for estimation, 77–81

- by analogy, 127–133
 - basic approach, 127–132
 - effort estimates, 209–210
 - estimating project size, 205
 - estimating schedule, 223–224
 - obtaining detailed information, 128
 - size comparisons, 129
 - uncertainty with, 128
- applicability tables, explained, 81
- computation, 83–84
 - converting counts to estimates, 86–88
 - stage-gate processes, 187
- counting, 84–86
 - computing estimates with, 86–88
 - function points, 200
 - stage-gate processes, 187
- Delphi technique, 150–155
- how to choose, 77–80
- judgment. *See* expert judgment; off-the-cuff estimates
- multiple approaches, using, 165–169
 - for effort, 218
 - for project size, 206
 - ranges, presenting, 256
 - for schedules, 231–232
 - stage-gate processes, 187
- proxy-based, 135–147
 - fuzzy logic, 136–138, 205
 - standard components approach, 138–141
 - story points, 142–144, 205
 - t-shirt sizing, 145–146
- software. *See* estimation software
- test for estimation skills, 15–19, 273
- tester-to-developer ratios, 237
- tightening estimates. *See* flow of estimates
- time measures, data on, 95, 97. *See also* schedules
- time periods, expressing, 255
- tool support, effect of, 65
- tools, estimation. *See* estimation software
- total activity, estimating, 235
- tracking accuracy of estimates, 111
- tracking progress
 - requirements changes and, 42
 - visibility of completion status, 27

U

uncertainty, expressing, 251–255. *See also*
 accuracy of estimates; accuracy of
 estimation method

underestimation, advantages of, 21–24

- unfamiliar environment, 247
- unforeseen events, 12
- unfounded optimism, 46, 107–108
 - avoiding with historical data, 93
- units of time estimates, 255
- unstable project requirements, 42, 247
 - requirements omitted from estimates, 44–46, 110
 - software to account for, 160
- unwarranted precision, 51–52
- urgency, instilling sense of, 22
- usefulness of estimates, need for, 13
- User Interface Design Complete phase, 35, 39. *See also* Cone of Uncertainty
- iterative development, 40–41

V

value of accurate estimates, 21–31

- benefits, 27–29
- other project attributes vs., 29–30
- overestimation vs. underestimation, 21–24
- problems with common techniques, 30

variability. *See* accuracy of estimates; accuracy of
 estimation method; Cone of Uncertainty

variance, 122

velocity (story points), 142

visibility of completion status, 27

vision estimates, 184

W

WBS (work breakdown structure), 117–132

Web pages, counting (example), 88

well-estimated projects, 172–173, 179–180

what-if analysis, 160

Wideband Delphi technique, 150–155

width of estimation ranges, 18–19. *See also* ranges
 of estimation

worst-case scenarios, 9, 40, 107–108. *See also*
 ranges of estimation

- adding to best case, 118–120
- diseconomies of scale, 58–59
- for multiple tasks, creating, 120–126
- presenting, 254

Steve McConnell

Steve McConnell is Chief Software Engineer at Construx Software where he oversees Construx's software engineering practices. Steve is the lead for the Construction Knowledge Area of the Software Engineering Body of Knowledge (SWEBOK) project. Steve has worked on software projects at Microsoft, Boeing, and other Seattle-area companies. Steve was the lead developer of Construx Estimate and of SPC Estimate Professional, winner of a *Software Development* magazine Productivity Award.

Steve is the author of *Rapid Development* (1996), *Software Project Survival Guide* (1998), *Professional Software Development* (2004) and *Code Complete, Second Edition* (2004). His books have twice won *Software Development* magazine's Jolt Product Excellence Award for outstanding software development book of the year. In 1998, readers of *Software Development* magazine named Steve one of the three most influential people in the software industry, along with Bill Gates and Linus Torvalds.

Steve earned a Bachelor's degree from Whitman College and a Master's degree in software engineering from Seattle University. He lives in Bellevue, Washington.

If you have any comments or questions about this book, please contact Steve at steve.mcconnell@construx.com or via www.stevemcconnell.com.



Additional Resources for Developers: Advanced Topics and Best Practices

Published and Forthcoming Titles from Microsoft Press

Code Complete, Second Edition

Steve McConnell • ISBN 0-7356-1967-0

For more than a decade, Steve McConnell, one of the premier authors and voices in the software community, has helped change the way developers write code—and produce better software. Now his classic book, *Code Complete*, has been fully updated and revised with best practices in the art and science of constructing software. Topics include design, applying good techniques to construction, eliminating errors, planning, managing construction activities, and relating personal character to superior software. This new edition features fully updated information on programming techniques, including the emergence of Web-style programming, and integrated coverage of object-oriented design. You'll also find new code examples—both good and bad—in C++, Microsoft® Visual Basic®, C#, and Java, although the focus is squarely on techniques and practices.

More About Software Requirements: Thorny Issues and Practical Advice

Karl E. Wiegers • ISBN 0-7356-2267-1

Have you ever delivered software that satisfied all of the project specifications, but failed to meet any of the customers expectations? Without formal, verifiable requirements—and a system for managing them—the result is often a gap between what developers think they're supposed to build and what customers think they're going to get. Too often, lessons about software requirements engineering processes are formal or academic, and not of value to real-world, professional development teams. In this follow-up guide to *Software Requirements*, Second Edition, you will discover even more practical techniques for gathering and managing software requirements that help you deliver software that meets project and customer specifications. Succinct and immediately useful, this book is a must-have for developers and architects.



Software Estimation: Demystifying the Black Art

Steve McConnell • ISBN 0-7356-0535-1

Often referred to as the “black art” because of its complexity and uncertainty, software estimation is not as hard or mysterious as people think. However, the art of how to create effective cost and schedule estimates has not been very well publicized. *Software Estimation* provides a proven set of procedures and heuristics that software developers, technical leads, and project managers can apply to their projects. Instead of arcane treatises and rigid modeling techniques, award-winning author Steve McConnell gives practical guidance to help organizations achieve basic estimation proficiency and lay the groundwork to continue improving project cost estimates. This book does not avoid the more complex mathematical estimation approaches, but the non-mathematical reader will find plenty of useful guidelines without getting bogged down in complex formulas.

Debugging, Tuning, and Testing Microsoft .NET 2.0 Applications

John Robbins • ISBN 0-7356-2202-7

Making an application the best it can be has long been a time-consuming task best accomplished with specialized and costly tools. With Microsoft Visual Studio® 2005, developers have available a new range of built-in functionality that enables them to debug their code quickly and efficiently, tune it to optimum performance, and test applications to ensure compatibility and trouble-free operation. In this accessible and hands-on book, debugging expert John Robbins shows developers how to use the tools and functions in Visual Studio to their full advantage to ensure high-quality applications.

The Security Development Lifecycle

Michael Howard and Steve Lipner • ISBN 0-7356-2214-0

Adapted from Microsoft's standard development process, the Security Development Lifecycle (SDL) is a methodology that helps reduce the number of security defects in code at every stage of the development process, from design to release. This book details each stage of the SDL methodology and discusses its implementation across a range of Microsoft software, including Microsoft Windows Server™ 2003, Microsoft SQL Server™ 2000 Service Pack 3, and Microsoft Exchange Server 2003 Service Pack 1, to help measurably improve security features. You get direct access to insights from Microsoft's security team and lessons that are applicable to software development processes worldwide, whether on a small-scale or a large-scale. This book includes a CD featuring videos of developer training classes.

Software Requirements, Second Edition

Karl E. Wiegers • ISBN 0-7356-1879-8

Writing Secure Code, Second Edition

Michael Howard and David LeBlanc • ISBN 0-7356-1722-8

CLR via C#, Second Edition

Jeffrey Richter • ISBN 0-7356-2163-2

For more information about Microsoft Press® books and other learning products, visit: www.microsoft.com/mspress and www.microsoft.com/learning

Microsoft
Press

Microsoft Press products are available worldwide wherever quality computer books are sold. For more information, contact your book or computer retailer, software reseller, or local Microsoft Sales Office, or visit our Web site at www.microsoft.com/mspress. To locate your nearest source for Microsoft Press products, or to order directly, call 1-800-MSPRESS in the United States. (In Canada, call 1-800-268-2222.)

Additional Resources for Web Developers

Published and Forthcoming Titles from Microsoft Press

Microsoft® Visual Web Developer™ 2005 Express Edition: Build a Web Site Now!

Jim Buyens • ISBN 0-7356-2212-4

With this lively, eye-opening, and hands-on book, all you need is a computer and the desire to learn how to create Web pages now using Visual Web Developer Express Edition! Featuring a full working edition of the software, this fun and highly visual guide walks you through a complete Web page project from set-up to launch. You'll get an introduction to the Microsoft Visual Studio® environment and learn how to put the lightweight, easy-to-use tools in Visual Web Developer Express to work right away—building your first, dynamic Web pages with Microsoft ASP.NET 2.0. You'll get expert tips, coaching, and visual examples at each step of the way, along with pointers to additional learning resources.

Microsoft ASP.NET 2.0 Programming Step by Step

George Shepherd • ISBN 0-7356-2201-9

With dramatic improvements in performance, productivity, and security features, Visual Studio 2005 and ASP.NET 2.0 deliver a simplified, high-performance, and powerful Web development experience. ASP.NET 2.0 features a new set of controls and infrastructure that simplify Web-based data access and include functionality that facilitates code reuse, visual consistency, and aesthetic appeal. Now you can teach yourself the essentials of working with ASP.NET 2.0 in the Visual Studio environment—one step at a time. With *Step by Step*, you work at your own pace through hands-on, learn-by-doing exercises. Whether you're a beginning programmer or new to this version of the technology, you'll understand the core capabilities and fundamental techniques for ASP.NET 2.0. Each chapter puts you to work, showing you how, when, and why to use specific features of the ASP.NET 2.0 rapid application development environment and guiding you as you create actual components and working applications for the Web, including advanced features such as personalization.

Programming Microsoft ASP.NET 2.0 Core Reference

Dino Esposito • ISBN 0-7356-2176-4

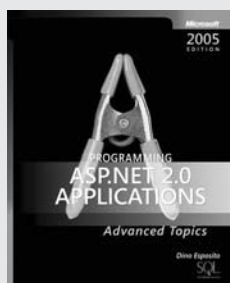
Delve into the core topics for ASP.NET 2.0 programming, mastering the essential skills and capabilities needed to build high-performance Web applications successfully. Well-known ASP.NET author Dino Esposito deftly builds your expertise with Web forms, Visual Studio, core controls, master pages, data access, data binding, state management, security services, and other must-know topics—combining definitive reference with practical, hands-on programming instruction. Packed with expert guidance and pragmatic examples, this *Core Reference* delivers the key resources that you need to develop professional-level Web programming skills.



Programming Microsoft ASP.NET 2.0 Applications: Advanced Topics

Dino Esposito • ISBN 0-7356-2177-2

Master advanced topics in ASP.NET 2.0 programming—gaining the essential insights and in-depth understanding that you need to build sophisticated, highly functional Web applications successfully. Topics include Web forms, Visual Studio 2005, core controls, master pages, data access, data binding, state management, and security considerations. Developers often discover that the more they use ASP.NET, the more they need to know. With expert guidance from ASP.NET authority Dino Esposito, you get the in-depth, comprehensive information that leads to full mastery of the technology.



Programming Microsoft Windows® Forms

Charles Petzold • ISBN 0-7356-2153-5

Programming Microsoft Web Forms

Douglas J. Reilly • ISBN 0-7356-2179-9

CLR via C++

Jeffrey Richter with Stanley B. Lippman
ISBN 0-7356-2248-5

Debugging, Tuning, and Testing Microsoft .NET 2.0 Applications

John Robbins • ISBN 0-7356-2202-7

CLR via C#, Second Edition

Jeffrey Richter • ISBN 0-7356-2163-2

For more information about Microsoft Press® books and other learning products, visit: www.microsoft.com/books and www.microsoft.com/learning

Microsoft®
Press

Microsoft Press products are available worldwide wherever quality computer books are sold. For more information, contact your book or computer retailer, software reseller, or local Microsoft Sales Office, or visit our Web site at www.microsoft.com/mspress. To locate your nearest source for Microsoft Press products, or to order directly, call 1-800-MSPRESS in the United States. (In Canada, call 1-800-268-2222.)

Additional Resources for C# Developers

Published and Forthcoming Titles from Microsoft Press

Microsoft® Visual C#® 2005 Express Edition: Build a Program Now!

Patrice Pelland • ISBN 0-7356-2229-9

In this lively, eye-opening, and hands-on book, all you need is a computer and the desire to learn how to program with Visual C# 2005 Express Edition. Featuring a full working edition of the software, this fun and highly visual guide walks you through a complete programming project—a desktop weather-reporting application—from start to finish. You'll get an unimposing introduction to the Microsoft Visual Studio® development environment and learn how to put the lightweight, easy-to-use tools in Visual C# Express to work right away—creating, compiling, testing, and delivering your first, ready-to-use program. You'll get expert tips, coaching, and visual examples at each step of the way, along with pointers to additional learning resources.

Microsoft Visual C# 2005 Step by Step

John Sharp • ISBN 0-7356-2129-2

Visual C#, a feature of Visual Studio 2005, is a modern programming language designed to deliver a productive environment for creating business frameworks and reusable object-oriented components. Now you can teach yourself essential techniques with Visual C#—and start building components and Microsoft Windows®-based applications—one step at a time. With *Step by Step*, you work at your own pace through hands-on, learn-by-doing exercises. Whether you're a beginning programmer or new to this particular language, you'll learn how, when, and why to use specific features of Visual C# 2005. Each chapter puts you to work, building your knowledge of core capabilities and guiding you as you create your first C#-based applications for Windows, data management, and the Web.

Programming Microsoft Visual C# 2005 Framework Reference

Francesco Balena • ISBN 0-7356-2182-9

Complementing *Programming Microsoft Visual C# 2005 Core Reference*, this book covers a wide range of additional topics and information critical to Visual C# developers, including Windows Forms, working with Microsoft ADO.NET 2.0 and Microsoft ASP.NET 2.0, Web services, security, remoting, and much more. Packed with sample code and real-world examples, this book will help developers move from understanding to mastery.

Programming Microsoft Visual C# 2005 Core Reference

Donis Marshall • ISBN 0-7356-2181-0

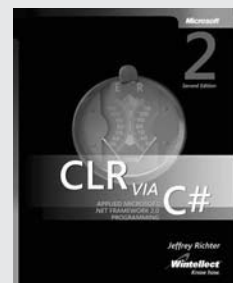
Get the in-depth reference and pragmatic, real-world insights you need to exploit the enhanced language features and core capabilities in Visual C# 2005. Programming expert Donis Marshall deftly builds your proficiency with classes, structs, and other fundamentals, and advances your expertise with more advanced topics such as debugging, threading, and memory management. Combining incisive reference with hands-on coding examples and best practices, this *Core Reference* focuses on mastering the C# skills you need to build innovative solutions for smart clients and the Web.



CLR via C#, Second Edition

Jeffrey Richter • ISBN 0-7356-2163-2

In this new edition of Jeffrey Richter's popular book, you get focused, pragmatic guidance on how to exploit the common language runtime (CLR) functionality in Microsoft .NET Framework 2.0 for applications of all types—from Web Forms, Windows Forms, and Web services to solutions for Microsoft SQL Server™, Microsoft code names "Avalon" and "Indigo," consoles, Microsoft Windows NT® Service, and more. Targeted to advanced developers and software designers, this book takes you under the covers of .NET for an in-depth understanding of its structure, functions, and operational components, demonstrating the most practical ways to apply this knowledge to your own development efforts. You'll master fundamental design tenets for .NET and get hands-on insights for creating high-performance applications more easily and efficiently. The book features extensive code examples in Visual C# 2005.



Programming Microsoft Windows Forms

Charles Petzold • ISBN 0-7356-2153-5

CLR via C++

Jeffrey Richter with Stanley B. Lippman
ISBN 0-7356-2248-5

Programming Microsoft Web Forms

Douglas J. Reilly • ISBN 0-7356-2179-9

Debugging, Tuning, and Testing Microsoft .NET 2.0 Applications

John Robbins • ISBN 0-7356-2202-7

For more information about Microsoft Press® books and other learning products, visit: www.microsoft.com/books and www.microsoft.com/learning

Microsoft
Press

Microsoft Press products are available worldwide wherever quality computer books are sold. For more information, contact your book or computer retailer, software reseller, or local Microsoft Sales Office, or visit our Web site at www.microsoft.com/mspress. To locate your nearest source for Microsoft Press products, or to order directly, call 1-800-MSPRESS in the United States. (In Canada, call 1-800-268-2222.)

Additional SQL Server Resources for Developers

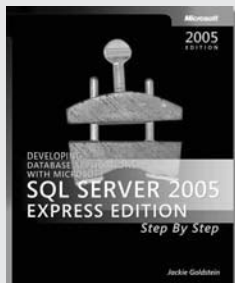
Published and Forthcoming Titles from Microsoft Press

Microsoft® SQL Server™ 2005 Express Edition *Step by Step*

Jackie Goldstein • ISBN 0-7356-2184-5

Teach yourself how to get database projects up and running quickly with SQL Server Express Edition—a free, easy-to-use database product that is based on SQL Server 2005 technology. It's designed for building simple, dynamic applications, with all the rich functionality of the SQL Server database engine and using the same data access APIs, such as Microsoft ADO.NET, SQL Native Client, and T-SQL.

Whether you're new to database programming or new to SQL Server, you'll learn how, when, and why to use specific features of this simple but powerful database development environment. Each chapter puts you to work, building your knowledge of core capabilities and guiding you as you create actual components and working applications.



Microsoft SQL Server 2005 Programming *Step by Step*

Fernando Guerrero • ISBN 0-7356-2207-8

SQL Server 2005 is Microsoft's next-generation data management and analysis solution that delivers enhanced scalability, availability, and security features to enterprise data and analytical applications while making them easier to create, deploy, and manage. Now you can teach yourself how to design, build, test, deploy, and maintain SQL Server databases—one step at a time. Instead of merely focusing on describing new features, this book shows new database programmers and administrators how to use specific features within typical business scenarios. Each chapter provides a highly practical learning experience that demonstrates how to build database solutions to solve common business problems.



Microsoft SQL Server 2005 Analysis Services *Step by Step*

Hitachi Consulting Services • ISBN 0-7356-2199-3

One of the key features of SQL Server 2005 is SQL Server Analysis Services—Microsoft's customizable analysis solution for business data modeling and interpretation. Just compare SQL Server Analysis Services to its competition to understand the great value of its enhanced features. One of the keys to harnessing the full functionality of SQL Server will be leveraging Analysis Services for the powerful tool that it is—including creating a cube, and deploying, customizing, and extending the basic calculations. This step-by-step tutorial discusses how to get started, how to build scalable analytical applications, and how to use and administer advanced features. Interactivity (enhanced in SQL Server 2005), data translation, and security are also covered in detail.

Microsoft SQL Server 2005 Reporting Services *Step by Step*

Hitachi Consulting Services • ISBN 0-7356-2250-7

SQL Server Reporting Services (SRS) is Microsoft's customizable reporting solution for business data analysis. It is one of the key value features of SQL Server 2005: functionality more advanced and much less expensive than its competition. SRS is powerful, so an understanding of how to architect a report, as well as how to install and program SRS, is key to harnessing the full functionality of SQL Server. This procedural tutorial shows how to use the Report Project Wizard, how to think about and access data, and how to build queries. It also walks through the creation of charts and visual layouts for maximum visual understanding of data analysis. Interactivity (enhanced in SQL Server 2005) and security are also covered in detail.

Programming Microsoft SQL Server 2005

Andrew J. Brust, Stephen Forte, and William H. Zack
ISBN 0-7356-1923-9

This thorough, hands-on reference for developers and database administrators teaches the basics of programming custom applications with SQL Server 2005. You will learn the fundamentals of creating database applications—including coverage of T-SQL, Microsoft .NET Framework, and Microsoft ADO.NET. In addition to practical guidance on database architecture and design, application development, and reporting and data analysis, this essential reference guide covers performance, tuning, and availability of SQL Server 2005.

Inside Microsoft SQL Server 2005: The Storage Engine

Kalen Delaney • ISBN 0-7356-2105-5

Inside Microsoft SQL Server 2005: T-SQL Programming

Itzik Ben-Gan • ISBN 0-7356-2197-7

Inside Microsoft SQL Server 2005: Query Processing and Optimization

Kalen Delaney • ISBN 0-7356-2196-9

Programming Microsoft ADO.NET 2.0 Core Reference

David Sceppe • ISBN 0-7356-2206-X

For more information about Microsoft Press® books and other learning products, visit: www.microsoft.com/mspress and www.microsoft.com/learning

Microsoft
Press

Microsoft Press products are available worldwide wherever quality computer books are sold. For more information, contact your book or computer retailer, software reseller, or local Microsoft Sales Office, or visit our Web site at www.microsoft.com/mspress. To locate your nearest source for Microsoft Press products, or to order directly, call 1-800-MSPRESS in the United States. (In Canada, call 1-800-268-2222.)

Additional Resources for Visual Basic Developers

Published and Forthcoming Titles from Microsoft Press

Microsoft® Visual Basic® 2005 Express Edition: Build a Program Now!

Patrice Pelland • ISBN 0-7356-2213-2

Featuring a full working edition of the software, this fun and highly visual guide walks you through a complete programming project—a desktop weather-reporting application—from start to finish. You'll get an introduction to the Microsoft Visual Studio® development environment and learn how to put the lightweight, easy-to-use tools in Visual Basic Express to work right away—creating, compiling, testing, and delivering your first ready-to-use program. You'll get expert tips, coaching, and visual examples each step of the way, along with pointers to additional learning resources.

Microsoft Visual Basic 2005 Step by Step

Michael Halvorson • ISBN 0-7356-2131-4

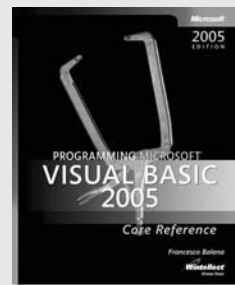
With enhancements across its visual designers, code editor, language, and debugger that help accelerate the development and deployment of robust, elegant applications across the Web, a business group, or an enterprise, Visual Basic 2005 focuses on enabling developers to rapidly build applications. Now you can teach yourself the essentials of working with Visual Studio 2005 and the new features of the Visual Basic language—one step at a time. Each chapter puts you to work, showing you how, when, and why to use specific features of Visual Basic and guiding as you create actual components and working applications for Microsoft Windows®. You'll also explore data management and Web-based development topics.



Programming Microsoft Visual Basic 2005 Core Reference

Francesco Balena • ISBN 0-7356-2183-7

Get the expert insights, indispensable reference, and practical instruction needed to exploit the core language features and capabilities in Visual Basic 2005. Well-known Visual Basic programming author Francesco Balena expertly guides you through the fundamentals, including modules, keywords, and inheritance, and builds your mastery of more advanced topics such as delegates, assemblies, and My Namespace. Combining in-depth reference with extensive, hands-on code examples and best-practices advice, this *Core Reference* delivers the key resources that you need to develop professional-level programming skills for smart clients and the Web.



Programming Microsoft Visual Basic 2005 Framework Reference

Francesco Balena • ISBN 0-7356-2175-6

Complementing *Programming Microsoft Visual Basic 2005 Core Reference*, this book covers a wide range of additional topics and information critical to Visual Basic developers, including Windows Forms, working with Microsoft ADO.NET 2.0 and ASP.NET 2.0, Web services, security, remoting, and much more. Packed with sample code and real-world examples, this book will help developers move from understanding to mastery.

Programming Microsoft Windows Forms

Charles Petzold • ISBN 0-7356-2153-5

Programming Microsoft Web Forms

Douglas J. Reilly • ISBN 0-7356-2179-9

Debugging, Tuning, and Testing Microsoft .NET 2.0 Applications

John Robbins • ISBN 0-7356-2202-7

Microsoft ASP.NET 2.0 Step by Step

George Shepherd • ISBN 0-7356-2201-9

Microsoft ADO.NET 2.0 Step by Step

Rebecca Riordan • ISBN 0-7356-2164-0

Programming Microsoft ASP.NET 2.0 Core Reference

Dino Esposito • ISBN 0-7356-2176-4

For more information about Microsoft Press® books and other learning products, visit: www.microsoft.com/books and www.microsoft.com/learning

Microsoft®
Press

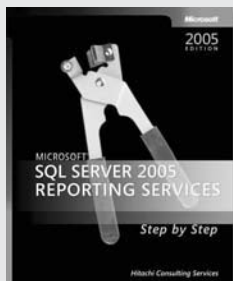
Additional SQL Server Resources for Administrators

Published and Forthcoming Titles from Microsoft Press

Microsoft® SQL Server™ 2005 Reporting Services *Step by Step*

Hitachi Consulting Services • ISBN 0-7356-2250-7

SQL Server Reporting Services (SRS) is Microsoft's customizable reporting solution for business data analysis. It is one of the key value features of SQL Server 2005: functionality more advanced and much less expensive than its competition. SRS is powerful, so an understanding of how to architect a report, as well as how to install and program SRS, is key to harnessing the full functionality of SQL Server. This procedural tutorial shows how to use the Report Project Wizard, how to think about and access data, and how to build queries. It also walks the reader through the creation of charts and visual layouts to enable maximum visual understanding of the data analysis. Interactivity (enhanced in SQL Server 2005) and security are also covered in detail.



Microsoft SQL Server 2005 Administrator's Pocket Consultant

William R. Stanek • ISBN 0-7356-2107-1

Here's the utterly practical, pocket-sized reference for IT professionals who need to administer, optimize, and maintain SQL Server 2005 in their organizations. This unique guide provides essential details for using SQL Server 2005 to help protect and manage your company's data—whether automating tasks; creating indexes and views; performing backups and recovery; replicating transactions; tuning performance; managing server activity; importing and exporting data; or performing other key tasks. Featuring quick-reference tables, lists, and step-by-step instructions, this handy, one-stop guide provides fast, accurate answers on the spot, whether you're at your desk or in the field!



Microsoft SQL Server 2005 Administrator's Companion

Marci Frohock Garcia, Edward Whalen, and Mitchell Schroeter • ISBN 0-7356-2198-5

Microsoft SQL Server 2005 Administrator's Companion is the comprehensive, in-depth guide that saves time by providing all the technical information you need to deploy, administer, optimize, and support SQL Server 2005. Using a hands-on, example-rich approach, this authoritative, one-volume reference book provides expert advice, product information, detailed solutions, procedures, and real-world troubleshooting tips from experienced SQL Server 2005 professionals. This expert guide shows you how to design high-availability database systems, prepare for installation, install and configure SQL Server 2005, administer services and features, and maintain and troubleshoot your database system. It covers how to configure your system for your I/O system and model and optimize system capacity. The expert authors provide details on how to create and use defaults, constraints, rules, indexes, views, functions, stored procedures, and triggers. This guide shows you how to administer reporting services, analysis services, notification services, and integration services. It also provides a wealth of information on replication and the specifics of snapshot, transactional, and merge replication. Finally, there is expansive coverage of how to manage and tune your SQL Server system, including automating tasks, backup and restoration of databases, and management of users and security.

Microsoft SQL Server 2005 Analysis Services *Step by Step*

Hitachi Consulting Services • ISBN 0-7356-2199-3

One of the key features of SQL Server 2005 is SQL Server Analysis Services—Microsoft's customizable analysis solution for business data modeling and interpretation. Just compare SQL Server Analysis Services to its competition to understand/grasp the great value of its enhanced features. One of the keys to harnessing the full functionality of SQL Server will be leveraging Analysis Services for the powerful tool that it is—including creating a cube, and deploying, customizing, and extending the basic calculations. This step-by-step tutorial discusses how to get started, how to build scalable analytical applications, and how to use and administer advanced features. Interactivity (which is enhanced in SQL Server 2005), data translation, and security are also covered in detail.

Microsoft SQL Server 2005 Express Edition *Step by Step*

Jackie Goldstein • ISBN 0-7356-2184-5

Inside Microsoft SQL Server 2005: The Storage Engine

Kalen Delaney • ISBN 0-7356-2105-5

Inside Microsoft SQL Server 2005: T-SQL Programming

Itzik Ben-Gan • ISBN 0-7356-2197-7

Inside Microsoft SQL Server 2005: Query Processing and Optimization

Kalen Delaney • ISBN 0-7356-2196-9

For more information about Microsoft Press® books and other learning products, visit: www.microsoft.com/mspress and www.microsoft.com/learning

Microsoft
Press

Microsoft Press products are available worldwide wherever quality computer books are sold. For more information, contact your book or computer retailer, software reseller, or local Microsoft Sales Office, or visit our Web site at www.microsoft.com/mspress. To locate your nearest source for Microsoft Press products, or to order directly, call 1-800-MSPRESS in the United States. (In Canada, call 1-800-268-2222.)

Additional Windows (R2) Resources for Administrators

Published and Forthcoming Titles from Microsoft Press

Microsoft® Windows Server™ 2003 Administrator's Pocket Consultant, Second Edition

William R. Stanek • ISBN 0-7356-2245-0

Here's the practical, pocket-sized reference for IT professionals supporting Microsoft Windows Server 2003—fully updated for Service Pack 1 and Release 2. Designed for quick referencing, this portable guide covers all the essentials for performing everyday system administration tasks. Topics include managing workstations and servers, using Active Directory® directory service, creating and administering user and group accounts, managing files and directories, performing data security and auditing tasks, handling data back-up and recovery, and administering networks using TCP/IP, WINS, and DNS, and more.



MCSE Self-Paced Training Kit (Exams 70-290, 70-291, 70-293, 70-294): Microsoft Windows Server 2003 Core Requirements, Second Edition

Holme, Thomas, Mackin, McLean, Zacker, Spealman, Hudson, and Craft • ISBN 0-7356-2290-6

The Microsoft Certified Systems Engineer (MCSE) credential is the premier certification for professionals who analyze the business requirements and design and implement the infrastructure for business solutions based on the Microsoft Windows Server 2003 platform and Microsoft Windows Server System—now updated for Windows Server 2003 Service Pack 1 and R2. This all-in-one set provides in-depth preparation for the four required networking system exams. Work at your own pace through the lessons, hands-on exercises, troubleshooting labs, and review questions. You get expert exam tips plus a full review section covering all objectives and sub-objectives in each study guide. Then use the Microsoft Practice Tests on the CD to challenge yourself with more than 1500 questions for self-assessment and practice!

MCSA/MCSE Self-Paced Training Kit (Exam 70-290): Managing and Maintaining a Microsoft Windows Server 2003 Environment, Second Edition

Dan Holme and Orin Thomas • ISBN 0-7356-2289-2

MCSA/MCSE Self-Paced Training Kit (Exam 70-291): Implementing, Managing, and Maintaining a Microsoft Windows Server 2003 Network Infrastructure, Second Edition

J.C. Mackin and Ian McLean • ISBN 0-7356-2288-4

Microsoft Windows® Small Business Server 2003 R2 Administrator's Companion

Charlie Russel, Sharon Crawford, and Jason Gerend • ISBN 0-7356-2280-9

Get your small-business network, messaging, and collaboration systems up and running quickly with the essential guide to administering Windows Small Business Server 2003 R2. This reference details the features, capabilities, and technologies for both the standard and premium editions—including Microsoft Windows Server 2003 R2, Exchange Server 2003 with Service Pack 1, Windows SharePoint® Services, SQL Server™ 2005 Workgroup Edition, and Internet Information Services. Discover how to install, upgrade, or migrate to Windows Small Business Server 2003 R2; plan and implement your network, Internet access, and security services; customize Microsoft Exchange Server for your e-mail needs; and administer user rights, shares, permissions, and Group Policy.



Microsoft Windows Small Business Server 2003 R2 Administrator's Companion

Charlie Russel, Sharon Crawford, and Jason Gerend • ISBN 0-7356-2280-9

Here's the ideal one-volume guide for the IT professional administering Windows Server 2003. Now fully updated for Windows Server 2003 Service Pack 1 and R2, this *Administrator's Companion* offers up-to-date information on core system administration topics for Microsoft Windows, including Active Directory services, security, scripting, disaster planning and recovery, and interoperability with UNIX. It also includes all-new sections on Service Pack 1 security updates and new features for R2. Featuring easy-to-use procedures and handy work-arounds, this book provides ready answers for on-the-job results.

MCSE Self-Paced Training Kit (Exam 70-293): Planning and Maintaining a Microsoft Windows Server 2003 Network Infrastructure, Second Edition

Craig Zacker • ISBN 0-7356-2287-6

MCSE Self-Paced Training Kit (Exam 70-294): Planning, Implementing, and Maintaining a Microsoft Windows Server 2003 Active Directory® Infrastructure, Second Edition

Jill Spealman, Kurt Hudson, and Melissa Craft • ISBN 0-7356-2286-8

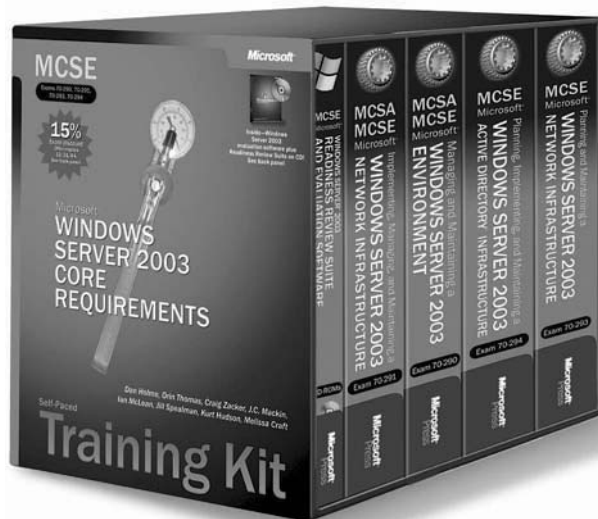
For more information about Microsoft Press® books and other learning products, visit: www.microsoft.com/mspress and www.microsoft.com/learning

Microsoft
Press

Microsoft Press products are available worldwide wherever quality computer books are sold. For more information, contact your book or computer retailer, software reseller, or local Microsoft Sales Office, or visit our Web site at www.microsoft.com/mspress. To locate your nearest source for Microsoft Press products, or to order directly, call 1-800-MSPRESS in the United States. (In Canada, call 1-800-268-2222.)

Prepare for Certification with Self-Paced Training Kits

Official Exam Prep Guides— Plus Practice Tests



Ace your preparation for the skills measured by the MCP exams—and on the job. With official *Self-Paced Training Kits* from Microsoft, you'll work at your own pace through a system of lessons, hands-on exercises, troubleshooting labs, and review questions. Then test yourself with the Readiness Review Suite on CD, which provides hundreds of challenging questions for in-depth self-assessment and practice.

- **MCSE Self-Paced Training Kit (Exams 70-290, 70-291, 70-293, 70-294): Microsoft® Windows Server™ 2003 Core Requirements.** 4-Volume Boxed Set. ISBN: 0-7356-1953-0. (Individual volumes are available separately.)
- **MCSA/MCSE Self-Paced Training Kit (Exam 70-270): Installing, Configuring, and Administering Microsoft Windows® XP Professional, Second Edition.** ISBN: 0-7356-2152-7.
- **MCSE Self-Paced Training Kit (Exam 70-298): Designing Security for a Microsoft Windows Server 2003 Network.** ISBN: 0-7356-1969-7.
- **MCSA/MCSE Self-Paced Training Kit (Exam 70-350): Implementing Microsoft Internet Security and Acceleration Server 2004.** ISBN: 0-7356-2169-1.
- **MCSA/MCSE Self-Paced Training Kit (Exam 70-284): Implementing and Managing Microsoft Exchange Server 2003.** ISBN: 0-7356-1899-2.

For more information about Microsoft Press® books, visit: www.microsoft.com/mspress

For more information about learning tools such as online assessments, e-learning, and certification, visit: www.microsoft.com/mspress and www.microsoft.com/learning

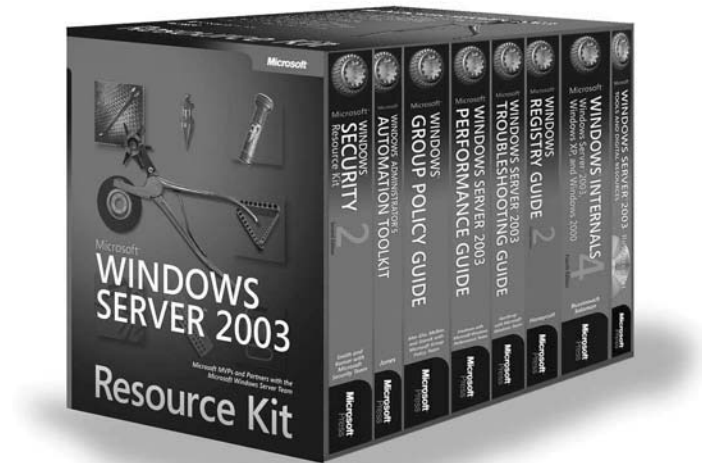
Microsoft®
Press

Microsoft Press products are available worldwide wherever quality computer books are sold. For more information, contact your book or computer retailer, software reseller, or local Microsoft Sales Office, or visit our Web site at www.microsoft.com/mspress. To locate your nearest source for Microsoft Press products, or to order directly, call 1-800-MSPRESS in the United States. (In Canada, call 1-800-268-2222.)

Microsoft Windows Server 2003 Resource Kit

The *definitive* resource

for Windows Server 2003!



Get the in-depth technical information and tools you need to manage and optimize Microsoft® Windows Server™ 2003—with expert guidance and best practices from Microsoft MVPs, leading industry consultants, and the Microsoft Windows Server team. This official *Resource Kit* delivers seven comprehensive volumes, including:

- **Microsoft Windows® Security Resource Kit, Second Edition**
- **Microsoft Windows Administrator's Automation Toolkit**
- **Microsoft Windows Group Policy Guide**
- **Microsoft Windows Server 2003 Performance Guide**
- **Microsoft Windows Server 2003 Troubleshooting Guide**
- **Microsoft Windows Registry Guide, Second Edition**
- **Microsoft Windows Internals, Fourth Edition**

You'll find 300+ timesaving tools and scripts, an eBook of the entire *Resource Kit*, plus five bonus eBooks. It's everything you need to help maximize system performance and reliability—and help reduce ownership and support costs.

Microsoft Windows Server 2003 Resource Kit

Microsoft MVPs and Partners with the Microsoft Windows Server Team

ISBN: 0-7356-2232-9

For more information about Microsoft Press® books, visit: www.microsoft.com/mspress

For more information about learning tools such as online assessments, e-learning, and certification, visit: www.microsoft.com/learning

Microsoft®
Press

Microsoft Press products are available worldwide wherever quality computer books are sold. For more information, contact your book or computer retailer, software reseller, or local Microsoft Sales Office, or visit our Web site at www.microsoft.com/mspress. To locate your nearest source for Microsoft Press products, or to order directly, call 1-800-MSPRESS in the United States. (In Canada, call 1-800-268-2222.)

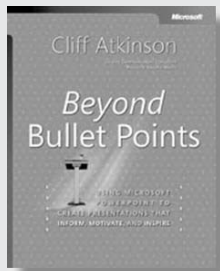
Additional Resources for Business and Home Users

Published and Forthcoming Titles from Microsoft Press

Beyond Bullet Points: Using Microsoft® PowerPoint® to Create Presentations That Inform, Motivate, and Inspire

Cliff Atkinson • ISBN 0-7356-2052-0

Improve your presentations—and increase your impact—with 50 powerful, practical, and easy-to-apply techniques for Microsoft PowerPoint. With *Beyond Bullet Points*, you'll take your presentation skills to the next level—learning innovative ways to design and deliver your message. Organized into five sections, including Distill Your Ideas, Structure Your Story, Visualize Your Message, Create a Conversation, and Maintain Engagement—the book uses clear, concise language and just the right visuals to help you understand concepts and start getting better results.



Take Back Your Life! Special Edition: Using Microsoft Outlook® to Get Organized and Stay Organized

Sally McGhee • ISBN 0-7356-2215-9

Unrelenting e-mail. Conflicting commitments. Endless interruptions. In this book, productivity expert Sally McGhee shows you how to take control and reclaim something that you thought you'd lost forever—your work-life balance. Now you can benefit from Sally's popular and highly regarded corporate education programs, learning simple but powerful techniques for rebalancing your personal and professional commitments using the productivity features in Outlook. When you change your approach, you can change your results. So learn what thousands of Sally's clients worldwide have discovered about taking control of their everyday productivity—and start transforming your own life today!

On Time! On Track! On Target! Managing Your Projects Successfully with Microsoft Project

Bonnie Biafore • ISBN 0-7356-2256-6

This book focuses on the core skills you need to successfully manage any project, giving you a practical education in project management and how-to instruction for using Microsoft Office Project Professional 2003 and other Microsoft Office Professional Edition 2003 programs, such as Excel® 2003, Outlook 2003, and Word 2003. Learn the essentials of project management, including creating successful project plans, tracking and evaluating performance, and controlling project costs. Whether you're a beginner just learning how to manage projects or a project manager already working on a project, this book has something for you. Includes a companion CD with sample Project templates.

Design to Sell: Using Microsoft Publisher to Inform, Motivate, and Persuade

Roger C. Parker • ISBN 0-7356-2260-4

Design to Sell relates the basics of effective message creation and formatting to the specific capabilities built into Microsoft Publisher—the powerful page layout program found on hundreds of thousands of computers around the world. Many Microsoft Office users already have Publisher on their computers but don't use it because they don't think of themselves as writers or designers. Here is a one-stop guide to marketing that even those without big budgets or previous design or writing experience can use to create compelling, easy-to-read marketing materials. Each chapter has an interactive exercise as well as questions with answers on the author's Web site. Also on the Web site are downloadable worksheets and templates, book updates, more illustrations of the projects in the book, and additional before-and-after project makeovers.

Microsoft Windows® XP Networking and Security Inside Out: Also Covers Windows 2000

Ed Bott and Carl Siechert • ISBN 0-7356-2042-3

Configure and manage your PC network—and help combat privacy and security threats—from the inside out! Written by the authors of the immensely popular *Microsoft Windows XP Inside Out*, this book packs hundreds of timesaving solutions, troubleshooting tips, and work-arounds for networking and security topics—all in concise, fast-answer format.



Dig into the tools and techniques for configuring workgroup, domain, Internet, and remote networking, and all the network components and features in between. Get the answers you need to use Windows XP Service Pack 2 and other tools, tactics, and features to help defend your personal computer and network against spyware, pop-up ads, viruses, hackers, spam, denial-of-service attacks, and other threats. Learn how to help secure your Virtual Private Networks (VPNs), remote access, and wireless networking services, and take ultimate control with advanced solutions such as file encryption, port blocking, IPSec, group policies, and tamper-proofing tactics for the registry. Get up to date on hot topics such as peer-to-peer networks, public wireless access points, smart cards, handheld computers, wireless LANs, and more. Plus, the CD includes bonus resources that make it easy for you to share your new security and networking expertise with your colleagues, friends, and family.

For more information about Microsoft Press® books and other learning products, visit: www.microsoft.com/mspress and www.microsoft.com/learning

Microsoft
Press

Microsoft Press products are available worldwide wherever quality computer books are sold. For more information, contact your book or computer retailer, software reseller, or local Microsoft Sales Office, or visit our Web site at www.microsoft.com/mspress. To locate your nearest source for Microsoft Press products, or to order directly, call 1-800-MSPRESS in the United States. (In Canada, call 1-800-268-2222.)