

"As a manager who worked under Michael Fisher and Marty Abbott during my time at PayPal/eBay, the opportunity to directly absorb the lessons and experiences presented in this book is invaluable to me now working at Facebook."

—**Yishan Wong**, former CEO, Reddit, and former director of engineering, Facebook



THE ART OF SCALABILITY

Scalable Web Architecture, Processes, and Organizations
for the Modern Enterprise

S E C O N D E D I T I O N

MARTIN L. ABBOTT

MICHAEL T. FISHER

FOREWORD BY **MARTY CAGAN**, FOUNDER, SILICON VALLEY PRODUCT GROUP

FREE SAMPLE CHAPTER

SHARE WITH OTHERS



Praise for *The Art of Scalability, Second Edition*

“A how-to manual for building a world-class engineering organization with step-by-step instructions on everything including leadership, architecture, operations, and processes. A driver’s manual for going from 0 to 60, scaling your business. With this book published, there’s no excuse for mistakes—in other words, RTFM.”

—Lon F. Binder, vice president, technology, Warby Parker

“I’ve worked with AKF for years on tough technical challenges. Many books address how to correct failing product architectures or problematic processes, both of which are symptoms of an unspoken problem. This book not only covers those symptoms, but also addresses their underlying cause—the way in which we manage, lead, organize, and staff our teams.”

—Jeremy King, chief technology officer and senior vice president,
global ecommerce, Walmart.com

“I love this book because it teaches an important lesson most technology-focused books don’t: how to build highly scalable and successful technology organizations that build highly scalable technology solutions. There’s plenty of great technology coaching in this book, but there are also excellent examples of how to build scalable culture, principles, processes, and decision trees. This book remains one of my few constant go-to reference guides.”

—Chris Schremser, chief technology officer, ZirMed

Praise for the First Edition

“This book is much more than you may think it is. Scale is not just about designing Web sites that don’t crash when lots of users show up. It is about designing your company so that it doesn’t crash when your business needs to grow. These guys have been there on the front lines of some of the most successful Internet companies of our time, and they share the good, the bad, and the ugly about how to not just survive, but thrive.”

—Marty Cagan, founder, Silicon Valley Product Group

“A must read for anyone building a Web service for the mass market.”

—Dana Stalder, general partner, Matrix Partners

“Abbott and Fisher have deep experiences with scale in both large and small enterprises. What’s unique about their approach to scalability is they start by focusing on the true foundation: people and process, without which true scalability cannot be built. Abbott and Fisher leverage their years of experience in a very accessible and practical approach to scalability that has been proven over time with their significant success.”

—Geoffrey Weber, vice president of internet operations/IT, Shutterfly

“If I wanted the best diagnoses for my health I would go to the Mayo Clinic. If I wanted the best diagnoses for my portfolio companies’ performance and scalability I would call Martin and Michael. They have recommended solutions to performance and scalability issues that have saved some of my companies from a total rewrite of the system.”

—Warren M. Weiss, general partner, Foundation Capital

“As a manager who worked under Michael Fisher and Marty Abbott during my time at PayPal/eBay, the opportunity to directly absorb the lessons and experiences presented in this book are invaluable to me now working at Facebook.”

—Yishan Wong, former CEO, Reddit, and former director of engineering, Facebook

“*The Art of Scalability* is by far the best book on scalability on the market today. The authors tackle the issues of scalability from processes, to people, to performance, to the highly technical. Whether your organization is just starting out and is defining processes as you go, or you are a mature organization, this is the ideal book

to help you deal with scalability issues before, during, or after an incident. Having built several projects, programs, and companies from small to significant scale, I can honestly say I wish I had this book one, five, and ten years ago.”

—Jeremy Wright, chief executive officer, b5media, Inc.

“Only a handful of people in the world have experienced the kind of growth-related challenges that Fisher and Abbott have seen at eBay, PayPal, and the other companies they’ve helped to build. Fewer still have successfully overcome such challenges. *The Art of Scalability* provides a great summary of lessons learned while scaling two of the largest internet companies in the history of the space, and it’s a must-read for any executive at a hyper-growth company. What’s more, it’s well-written and highly entertaining. I couldn’t put it down.”

—Kevin Fortuna, partner, AKF Consulting

“Marty and Mike’s book covers all the bases, from understanding how to build a scalable organization to the processes and technology necessary to run a highly scalable architecture. They have packed in a ton of great practical solutions from real world experiences. This book is a must-read for anyone having difficulty managing the scale of a hyper-growth company or a startup hoping to achieve hyper growth.”

—Tom Keeven, partner, AKF Consulting

“*The Art of Scalability* is remarkable in its wealth of information and clarity; the authors provide novel, practical, and demystifying approaches to identify, predict, and resolve scalability problems before they surface. Marty Abbott and Michael Fisher use their rich experience and vision, providing unique and groundbreaking tools to assist small and hyper-growth organizations as they maneuver in today’s demanding technological environments.”

—Joseph M. Potenza, attorney, Banner & Witcoff, Ltd.

This page intentionally left blank

The Art of Scalability

Second Edition

This page intentionally left blank

The Art of Scalability

*Scalable Web Architecture, Processes,
and Organizations for the Modern
Enterprise*

Second Edition

Martin L. Abbott

Michael T. Fisher

◆ Addison-Wesley

New York • Boston • Indianapolis • San Francisco
Toronto • Montreal • London • Munich • Paris • Madrid
Capetown • Sydney • Tokyo • Singapore • Mexico City

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and the publisher was aware of a trademark claim, the designations have been printed with initial capital letters or in all capitals.

The authors and publisher have taken care in the preparation of this book, but make no expressed or implied warranty of any kind and assume no responsibility for errors or omissions. No liability is assumed for incidental or consequential damages in connection with or arising out of the use of the information or programs contained herein.

For information about buying this title in bulk quantities, or for special sales opportunities (which may include electronic versions; custom cover designs; and content particular to your business, training goals, marketing focus, or branding interests), please contact our corporate sales department at corpsales@pearsoned.com or (800) 382-3419.

For government sales inquiries, please contact governmentsales@pearsoned.com.

For questions about sales outside the U.S., please contact international@pearsoned.com.

Visit us on the Web: informit.com/aw

Library of Congress Cataloging-in-Publication Data

Abbott, Martin L.

The art of scalability : scalable web architecture, processes, and organizations for the modern enterprise / Martin L. Abbott, Michael T. Fisher.

pages cm

Includes index.

ISBN 978-0-13-403280-1 (pbk. : alk. paper)

1. Web site development. 2. Computer networks—Scalability. 3. Business enterprises—Computer networks. I. Fisher, Michael T. II. Title.

TK5105.888.A2178 2015

658.4'06—dc23

2015009317

Copyright © 2015 Pearson Education, Inc.

All rights reserved. Printed in the United States of America. This publication is protected by copyright, and permission must be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. To obtain permission to use material from this work, please submit a written request to Pearson Education, Inc., Permissions Department, 200 Old Tappan Road, Old Tappan, New Jersey 07675, or you may fax your request to (201) 236-3290.

ISBN-13: 978-0-13-403280-1

ISBN-10: 0-13-403280-2

Text printed in the United States on recycled paper at RR Donnelley in Crawfordsville, Indiana.

First printing, June 2015

Editor-in-Chief

Mark L. Taub

Executive Editor

Laura Lewin

Development Editor

Songlin Qiu

Managing Editor

John Fuller

Senior Project

Editor

Mary Kesel Wilson

Copy Editor

Jill Hobbs

Indexer

Jack Lewis

Proofreader

Andrea Fox

Technical Reviewers

Roger Andelin

Chris Schremser

Geoffrey Weber

Editorial Assistant

Olivia Basegio

Cover Designer

Chuti Prasertsith

Compositor

The CIP Group

*“To my father, for teaching me how to succeed, and to my wife
Heather, for teaching me how to have fun.”*

—Marty Abbott

*“To my parents, for their guidance, and to my wife and son, for their
unflagging support.”*

—Michael Fisher

This page intentionally left blank

Contents

Foreword	xxiii
Acknowledgments	xxvii
About the Authors	xxix
Introduction	1
Part I: Staffing a Scalable Organization.	7
Chapter 1: The Impact of People and Leadership on Scalability.	9
The Case Method	9
Why People?	10
Why Organizations?	11
Why Management and Leadership?	17
Conclusion	19
Key Points.	20
Chapter 2: Roles for the Scalable Technology Organization.	21
The Effects of Failure	21
Defining Roles	23
Executive Responsibilities.	25
Chief Executive Officer	25
Chief Financial Officer	27
Business Unit Owners, General Managers, and P&L Owners	27
Chief Technology Officer/Chief Information Officer	28
Individual Contributor Responsibilities	30
Architecture Responsibilities	30
Engineering Responsibilities.	31
DevOps Responsibilities.	31
Infrastructure Responsibilities	32
Quality Assurance Responsibilities.	33
Capacity Planning Responsibilities.	33

A Tool for Defining Responsibilities	35
Conclusion	39
Key Points	40
Chapter 3: Designing Organizations	41
Organizational Influences That Affect Scalability	41
Team Size	44
Warning Signs	47
Growing or Splitting Teams	48
Organizational Structure	51
Functional Organization	51
Matrix Organization	56
Agile Organization	59
Conclusion	69
Key Points	70
Chapter 4: Leadership 101	71
What Is Leadership?	72
Leadership: A Conceptual Model	74
Taking Stock of Who You Are	76
Leading from the Front	78
Checking Your Ego at the Door	79
Mission First, People Always	80
Making Timely, Sound, and Morally Correct Decisions	81
Empowering Teams and Scalability	82
Alignment with Shareholder Value	83
Transformational Leadership	84
Vision	84
Mission	87
Goals	89
Putting It All Together	90
The Causal Roadmap to Success	94
Conclusion	95
Key Points	96
Chapter 5: Management 101	99
What Is Management?	100
Project and Task Management	102
Building Teams: A Sports Analogy	105

Upgrading Teams: A Garden Analogy	107
Measurement, Metrics, and Goal Evaluation	111
The Goal Tree	114
Paving the Path for Success	115
Conclusion	116
Key Points	117
Chapter 6: Relationships, Mindset, and the Business Case	119
Understanding the Experiential Chasm	119
Why the Business Executive Might Be the Problem	120
Why the Technology Executive Might Be the Problem	121
Defeating the IT Mindset	122
The Business Case for Scale	124
Conclusion	127
Key Points	128
Part II: Building Processes for Scale	129
Chapter 7: Why Processes Are Critical to Scale	131
The Purpose of Process	132
Right Time, Right Process	135
A Process Maturity Framework	136
When to Implement Processes	137
Process Complexity	137
When Good Processes Go Bad	139
Conclusion	140
Key Points	141
Chapter 8: Managing Incidents and Problems	143
What Is an Incident?	144
What Is a Problem?	145
The Components of Incident Management	146
The Components of Problem Management	149
Resolving Conflicts Between Incident and Problem Management	150
Incident and Problem Life Cycles	150
Implementing the Daily Incident Meeting	152
Implementing the Quarterly Incident Review	153
The Postmortem Process	153

Putting It All Together	156
Conclusion	157
Key Points.	158
Chapter 9: Managing Crises and Escalations	159
What Is a Crisis?	160
Why Differentiate a Crisis from Any Other Incident?	161
How Crises Can Change a Company	162
Order Out of Chaos	163
The Role of the Problem Manager	164
The Role of Team Managers.	166
The Role of Engineering Leads.	167
The Role of Individual Contributors.	167
Communications and Control.	168
The War Room	169
Escalations	170
Status Communications	171
Crisis Postmortem and Communication	172
Conclusion	173
Key Points.	174
Chapter 10: Controlling Change in Production Environments	177
What Is a Change?	178
Change Identification	179
Change Management	180
Change Proposal.	183
Change Approval	186
Change Scheduling.	187
Change Implementation and Logging.	189
Change Validation	189
Change Review.	191
The Change Control Meeting	191
Continuous Process Improvement.	192
Conclusion	193
Key Points.	194
Chapter 11: Determining Headroom for Applications	197
Purpose of the Process	198
Structure of the Process	199
Ideal Usage Percentage	203

A Quick Example Using Spreadsheets	206
Conclusion	207
Key Points	208
Chapter 12: Establishing Architectural Principles	209
Principles and Goals	209
Principle Selection	212
AKF's Most Commonly Adopted Architectural Principles	214
N + 1 Design	214
Design for Rollback	215
Design to Be Disabled	215
Design to Be Monitored	215
Design for Multiple Live Sites	216
Use Mature Technologies	217
Asynchronous Design	217
Stateless Systems	218
Scale Out, Not Up	219
Design for at Least Two Axes of Scale	219
Buy When Non-Core	220
Use Commodity Hardware	220
Build Small, Release Small, Fail Fast	221
Isolate Faults	221
Automation over People	221
Conclusion	222
Key Points	223
Chapter 13: Joint Architecture Design and Architecture Review Board	225
Fixing Organizational Dysfunction	225
Designing for Scale Cross-Functionally	226
JAD Entry and Exit Criteria	228
From JAD to ARB	230
Conducting the Meeting	232
ARB Entry and Exit Criteria	234
Conclusion	236
Key Points	237
Chapter 14: Agile Architecture Design	239
Architecture in Agile Organizations	240
Ownership of Architecture	241
Limited Resources	242

Standards	243
ARB in the Agile Organization	246
Conclusion	247
Key Points	247
Chapter 15: Focus on Core Competencies: Build Versus Buy	249
Building Versus Buying, and Scalability	249
Focusing on Cost	250
Focusing on Strategy	251
“Not Built Here” Phenomenon	252
Merging Cost and Strategy	252
Does This Component Create Strategic Competitive Differentiation?.	253
Are We the Best Owners of This Component or Asset?	253
What Is the Competition for This Component?	254
Can We Build This Component Cost-Effectively?	254
The Best Buy Decision Ever	255
Anatomy of a Build-It-Yourself Failure	256
Conclusion	258
Key Points	258
Chapter 16: Determining Risk	259
Importance of Risk Management to Scale	259
Measuring Risk	261
Managing Risk	268
Conclusion	271
Key Points	272
Chapter 17: Performance and Stress Testing	273
Performing Performance Testing	273
Establish Success Criteria	274
Establish the Appropriate Environment	275
Define the Tests	276
Execute the Tests	277
Analyze the Data	278
Report to Engineers	279
Repeat the Tests and Analysis	279
Don’t Stress over Stress Testing	281
Identify the Objectives	281
Identify the Key Services	282
Determine the Load	283

Establish the Appropriate Environment	283
Identify the Monitors	284
Create the Load	284
Execute the Tests	284
Analyze the Data	285
Performance and Stress Testing for Scalability	287
Conclusion	288
Key Points.	289
Chapter 18: Barrier Conditions and Rollback	291
Barrier Conditions	291
Barrier Conditions and Agile Development	293
Barrier Conditions and Waterfall Development	295
Barrier Conditions and Hybrid Models	296
Rollback Capabilities	297
Rollback Window	297
Rollback Technology Considerations	298
Cost Considerations of Rollback	299
Markdown Functionality: Design to Be Disabled	300
Conclusion	301
Key Points.	302
Chapter 19: Fast or Right?	303
Tradeoffs in Business	303
Relation to Scalability.	306
How to Think About the Decision	307
Conclusion	311
Key Points.	313
Part III: Architecting Scalable Solutions	315
Chapter 20: Designing for Any Technology	317
An Implementation Is Not an Architecture.	317
Technology-Agnostic Design.	318
TAD and Cost	319
TAD and Risk	320
TAD and Scalability.	321
TAD and Availability	323
The TAD Approach	323

Conclusion	325
Key Points.	325
Chapter 21: Creating Fault-Isolative Architectural Structures	327
Fault-Isolative Architecture Terms	327
Benefits of Fault Isolation	329
Fault Isolation and Availability: Limiting Impact	329
Fault Isolation and Availability: Incident Detection and Resolution	334
Fault Isolation and Scalability	334
Fault Isolation and Time to Market	334
Fault Isolation and Cost	335
How to Approach Fault Isolation	336
Principle 1: Nothing Is Shared	337
Principle 2: Nothing Crosses a Swim Lane Boundary.	338
Principle 3: Transactions Occur Along Swim Lanes	338
When to Implement Fault Isolation.	339
Approach 1: Swim Lane the Money-Maker	339
Approach 2: Swim Lane the Biggest Sources of Incidents.	339
Approach 3: Swim Lane Along Natural Barriers	340
How to Test Fault-Isolative Designs	341
Conclusion	341
Key Points.	342
Chapter 22: Introduction to the AKF Scale Cube	343
The AKF Scale Cube.	343
The <i>x</i> -Axis of the Cube.	344
The <i>y</i> -Axis of the Cube.	346
The <i>z</i> -Axis of the Cube.	349
Putting It All Together	350
When and Where to Use the Cube	352
Conclusion	353
Key Points.	354
Chapter 23: Splitting Applications for Scale	357
The AKF Scale Cube for Applications	357
The <i>x</i> -Axis of the AKF Application Scale Cube	359
The <i>y</i> -Axis of the AKF Application Scale Cube	361
The <i>z</i> -Axis of the AKF Application Scale Cube	363
Putting It All Together	365

Practical Use of the Application Cube	367
Observations	370
Conclusion	371
Key Points	372
Chapter 24: Splitting Databases for Scale	375
Applying the AKF Scale Cube to Databases	375
The <i>x</i> -Axis of the AKF Database Scale Cube	376
The <i>y</i> -Axis of the AKF Database Scale Cube	381
The <i>z</i> -Axis of the AKF Database Scale Cube	383
Putting It All Together	385
Practical Use of the Database Cube	388
Ecommerce Implementation	388
Search Implementation	389
Business-to-Business SaaS Solution	391
Observations	392
Timeline Considerations	393
Conclusion	393
Key Points	394
Chapter 25: Caching for Performance and Scale	395
Caching Defined	395
Object Caches	399
Application Caches	402
Proxy Caches	402
Reverse Proxy Caches	403
Caching Software	406
Content Delivery Networks	407
Conclusion	408
Key Points	409
Chapter 26: Asynchronous Design for Scale	411
Synching Up on Synchronization	411
Synchronous Versus Asynchronous Calls	412
Scaling Synchronously or Asynchronously	414
Example Asynchronous Systems	416
Defining State	418
Conclusion	422
Key Points	423

Part IV: Solving Other Issues and Challenges	425
Chapter 27: Too Much Data	427
The Cost of Data	427
The Value of Data and the Cost-Value Dilemma	430
Making Data Profitable	431
Option Value	431
Strategic Competitive Differentiation	432
Cost-Justify the Solution (Tiered Storage Solutions)	432
Transform the Data	433
Handling Large Amounts of Data	434
Big Data	438
A NoSQL Primer	440
Conclusion	444
Key Points	444
Chapter 28: Grid Computing	447
History of Grid Computing	447
Pros and Cons of Grids	449
Pros of Grids	449
Cons of Grids	452
Different Uses for Grid Computing	454
Production Grid	454
Build Grid	455
Data Warehouse Grid	456
Back-Office Grid	456
Conclusion	457
Key Points	458
Chapter 29: Soaring in the Clouds	459
History and Definitions	460
Public Versus Private Clouds	462
Characteristics and Architecture of Clouds	463
Pay by Usage	463
Scale on Demand	464
Multiple Tenants	465
Virtualization	466
Differences Between Clouds and Grids	467

Pros and Cons of Cloud Computing	468
Pros of Cloud Computing	468
Cons of Cloud Computing	471
Where Clouds Fit in Different Companies	476
Environments	476
Skill Sets	478
Decision Process	478
Conclusion	481
Key Points.	482
Chapter 30: Making Applications Cloud Ready	485
The Scale Cube in a Cloud	485
<i>x</i> -Axis.	485
<i>y</i> - and <i>z</i> -Axes	486
Overcoming Challenges	487
Fault Isolation in a Cloud	487
Variability in Input/Output	489
Intuit Case Study	491
Conclusion	493
Key Points.	494
Chapter 31: Monitoring Applications	495
“Why Didn’t We Catch That Earlier?”	495
A Framework for Monitoring	496
User Experience and Business Metrics	499
Systems Monitoring	501
Application Monitoring	502
Measuring Monitoring: What Is and Isn’t Valuable?.	503
Monitoring and Processes.	504
Conclusion	506
Key Points.	507
Chapter 32: Planning Data Centers	509
Data Center Costs and Constraints	509
Location, Location, Location	511
Data Centers and Incremental Growth.	514
When Do I Consider IaaS?	516
Three Magic Rules of Three	519
The First Rule of Three: Three Magic Drivers of Data Center Costs.	520



- The Second Rule of Three: Three Is the Magic Number
for Servers 520
- The Third Rule of Three: Three Is the Magic Number for
Data Centers 521
- Multiple Active Data Center Considerations. 525
- Conclusion 527
- Key Points. 528
- Chapter 33: Putting It All Together 531**
 - What to Do Now?..... 532
 - Further Resources on Scalability..... 535
 - Blogs..... 535
 - Books 535
- Part V: Appendices 537**
 - Appendix A: Calculating Availability 539**
 - Hardware Uptime..... 540
 - Customer Complaints..... 541
 - Portion of Site Down..... 542
 - Third-Party Monitoring Service 543
 - Business Graph 544
 - Appendix B: Capacity Planning Calculations..... 547**
 - Appendix C: Load and Performance Calculations 555**
 - Index 563**

Foreword

Perhaps your company began as a brick-and-mortar retailer, or an airline, or a financial services company.

A retailer creates (or buys) technology to coordinate and manage inventory, distribution, billing, and point of sale systems. An airline creates technology to manage the logistics involved in flights, crews, reservations, payment, and fleet maintenance. A financial services company creates technology to manage its customers' assets and investments.

But over the past several years, almost all of these companies, as well as their counterparts in nearly every other industry, have realized that to remain competitive, they need to take their use of technology to an entirely different level—they now need to engage directly with their customers.

Every industry is being reshaped by technology. If they hope to maintain their place as competitive, viable enterprises, companies have no choice but to embrace technology, often in ways that go well beyond their comfort zone.

For example, most retailers now find they need to sell their goods directly to consumers online. Most airlines are trying very hard to entice their customers to purchase their air travel online directly through the airline's site. And nearly all financial services companies work to enable their customers to manage assets and trade directly via their real-time financial sites.

Unfortunately, many of these companies are trying to manage this new customer-facing and customer-enabling technology in the same way they manage their internal technology. The result is that many of these companies have very broken technology and provide terrible customer experiences. Even worse, they don't have the organization, people, or processes in place to improve them.

What companies worldwide are discovering is that there is a very profound difference between utilizing technology to help *run* your company, and leveraging technology to provide your actual products and services directly *for* your customers. It also explains why "technology transformation" initiatives are popping up at so many companies.

This book is all about this necessary transformation. Such a transformation represents a shift in organization, people, process, and especially culture, and scalability is at the center of this transformation.

- Scaling from hundreds of your employees using your technology, to millions of your customers depending on your technology

- Scaling from a small IT cost-center team serving their colleagues in finance and marketing, to a substantial profit-center technology team serving your customers
- More generally, scaling your people, processes, and technology to meet the demands of a modern technology-powered business

But why is technology for your customers so different and so much more difficult to manage than technology for your employees? Several reasons:

- You pay your employees to work at your company and use the technology you tell them they need to use. In contrast, every customer makes his or her own purchase decision—and if she doesn't want it, she won't use it. Your customers must *choose* to use your technology.
- With your own employees, you can get away with requiring training courses, reading manuals, or holding their hands if necessary. In contrast, if your customers can't figure out how to use your technology, they are just a click away from your competitor.
- For internal technology, we measure scale and simultaneous usage in the hundreds of users. For our customers, that scope increases to hundreds of thousands or very often millions of users.
- With internal technology, if a problem arises with the technology, the users are your employees and they are forced to deal with it. For your customers, an issue such as an outage immediately disrupts revenue streams, usually gets the attention of the CEO, and sometimes even draws the notice of the press.
- The harsh truth is that most customer technology simply has a dramatically higher bar set in terms of the definition, design, implementation, testing, deployment, and support than is necessary with most internal technology.

For most companies, establishing a true customer technology competency is *the single most important thing for them to be doing to ensure their survival*, yet remarkably some of them don't even realize they have a problem. They assume that “technology is technology” and the same people who managed their enterprise resource planning implementation shouldn't have too much trouble getting something going online.

If your company is in need of this transformation, then this book is essential reading. It provides a proven blueprint for the necessary change.

Marty and Michael have been there and done that with most of the technology industry's leading companies. I have known and worked with both of these guys for many years. They are not management consultants who could barely launch a

brochure site. They are hands-on leaders who have spent decades in the trenches with their teams creating technology-powered businesses serving hundreds of millions of users and customers. They are the best in the world at what they do, and this new edition is a goldmine of information for any technology organization working to raise its game.

—*Marty Cagan*
Founder, Silicon Valley Product Group

This page intentionally left blank

Acknowledgments

The authors would like to recognize, first and foremost, the experience and advice of our partner and cofounder Tom Keeven. The process and technology portions of this book were built over time with the help of Tom and his many years of experience. Tom started the business that became AKF Partners. We often joke that Tom has forgotten more about architecting highly available and scalable sites than most of us will ever learn.

We would also like to thank several AKF team members—Geoff Kershner, Dave Berardi, Mike Paylor, Kirk Sanford, Steve Mason, and Alex Hooper—who contributed their combined decades of experience and knowledge not only to this second edition, but also to AKF Partners’ consulting practice. Without their help putting the concepts from the first edition into practice and helping to mature them over time, this second edition would not be possible.

Additionally, the authors owe a great debt of gratitude to this edition’s technical reviewers—Geoffrey Weber, Chris Schremser, and Roger Andelin. All three of these individuals are experienced technology executives who have decades of hands-on experience designing, developing, implementing, and supporting large-scale systems in industries ranging from ecommerce to health care. They willingly agreed to accept our poorly written drafts and help turn them into easily consumable prose for the benefit of our readers.

This edition would not be possible without the support provided by the team at Addison-Wesley, including executive editor Laura Lewin, development editor Songlin Qiu, and editorial assistant Olivia Basegio. Laura quickly became the champion for a second edition after discussing the significant changes with regard to scaling systems and organizations that have occurred over the five years since the first edition was published. Songlin has been an invaluable partner in ensuring both the first and second editions of *The Art of Scalability* were consistent, clear, and correct. Olivia has saved us multiple times when technical challenges threatened to delay or derail us.

We further would like to recognize our colleagues and teams at Quigo, eBay, and PayPal. These are the companies at which we really started to build and test many of the approaches mentioned in the technology and process sections of this book. The list of names within these teams is quite large, but the individuals know who they are.

Finally, we’d like to acknowledge the U.S. Army and United States Military Academy. Together they created a leadership lab unlike any other we can imagine.

Multiple reviewers have reviewed this book as we have attempted to provide the best possible work for the reader. However, in a work this large, errors will inevitably occur. All errors in the text are completely the authors’ fault.

This page intentionally left blank

About the Authors

Martin L. Abbott is a founding partner at the growth and scalability advisory firm AKF Partners. He was formerly chief operations officer at Quigo, an advertising technology startup sold to AOL, where he was responsible for product strategy, product management, technology development, and client services. Marty spent nearly six years at eBay, most recently as senior vice president of technology, chief technology officer, and member of the executive staff. Prior to his time at eBay, Marty held domestic and international engineering, management, and executive positions at Gateway and Motorola. He has served on the boards of directors of several private and public companies. Marty has a B.S. in computer science from the United States Military Academy, has an M.S. in computer engineering from the University of Florida, is a graduate of the Harvard Business School Executive Education Program, and has a Doctor of Management from Case Western Reserve University.

Michael T. Fisher is a founding partner at the growth and scalability advisory firm AKF Partners. Prior to cofounding AKF Partners, Michael was the chief technology officer at Quigo, a startup Internet advertising company that was acquired by AOL in 2007. Before his time at Quigo, Michael served as vice president, engineering and architecture, for PayPal, Inc., an eBay company. Prior to joining PayPal, he spent seven years at General Electric helping to develop the company's technology strategy and was a Six Sigma Master Black Belt. Michael served six years as a Captain and pilot in the U.S. Army. He received a Ph.D. and an MBA from Case Western Reserve University's Weatherhead School of Management, an M.S. in information systems from Hawaii-Pacific University, and a B.S. in computer science from the United States Military Academy (West Point). Michael is an adjunct professor in the design and innovation department at Case Western Reserve University's Weatherhead School of Management.

This page intentionally left blank

Introduction

Thanks for picking up the second edition of *The Art of Scalability: Scalable Web Architecture, Processes, and Organizations for the Modern Enterprise*. This book has been recognized by academics and professionals as one of the best resources available to learn the art of scaling systems and organizations. This second edition includes new content, revisions, and updates. As consultants and advisors to hundreds of hyper-growth companies, we have been fortunate enough to be on the forefront of many industry changes, including new technologies and new approaches to implementing products. While we hope our clients see value in our knowledge and experience, we are not ignorant of the fact that a large part of the value we bring to bear on a subject comes from our interactions with so many other technology companies. In this edition, we share even more of these lessons learned from our consulting practice.

In this second edition, we have added several key topics that we believe are critical to address in a book on scalability. One of the most important new topics focuses on a new organizational structure that we refer to as the Agile Organization. Other notable topics include the changing rationale for moving from data centers to clouds (IaaS/PaaS), why NoSQL solutions aren't in and of themselves a panacea for scaling, and the importance of business metrics to the health of the overall system.

In the first edition of *The Art of Scalability*, we used a fictional company called AllScale to demonstrate many of the concepts. This fictional company was an aggregation of many of our clients and the challenges they faced in the real world. While AllScale provided value in highlighting the key points in the first edition, we believe that real stories make more of an impact with readers. As such, we've replaced AllScale with real-world stories of successes and failures in the current edition.

The information contained in this book has been carefully designed to be appropriate for any employee, manager, or executive of an organization or company that provides technology solutions. For the nontechnical executive or product manager, this book can help you prevent scalability disasters by arming you with the tools needed to ask the right questions and focus on the right areas. For technologists and engineers, this book provides models and approaches that, once employed, will help you scale your products, processes, and organizations.

Our experience with scalability goes beyond academic study and research. Although we are both formally trained as engineers, we don't believe academic

programs teach scalability very well. Rather, we have learned about scalability by suffering through the challenges of scaling systems for a combined 30-plus years. We have been engineers, managers, executives, and advisors for startups as well as *Fortune 500* companies. The list of companies that our firm or we as individuals have worked with includes such familiar names as General Electric, Motorola, Gateway, eBay, Intuit, Salesforce, Apple, Dell, Walmart, Visa, ServiceNow, DreamWorks Animation, LinkedIn, Carbonite, Shutterfly, and PayPal. The list also includes hundreds of less famous startups that need to be able to scale as they grow. Having learned the scalability lessons through thousands of hours spent diagnosing problems and thousands more hours spent designing preventions for those problems, we want to share our combined knowledge. This motivation was the driving force behind our decisions to start our consulting practice, AKF Partners, in 2007, and to write the first edition of this book, and it remains our preeminent goal in this second edition.

Scalability: So Much More Than Just Technology

Pilots are taught, and statistics show, that many aircraft incidents are the result of multiple failures that snowball into total system failure and catastrophe. In aviation, these multiple failures, which are called an error chain, often start with human rather than mechanical failure. In fact, Boeing identified that 55% of all aircraft incidents involving Boeing aircraft between 1995 and 2005 had human factors-related causes.¹

Our experience with scalability-related issues follows a similar trend. The chief technology officer (CTO) or executive responsible for scale of a technology platform may see scalability as purely a technical endeavor. This perception is the first, and very human, failure in the error chain. Because the CTO is overly technology focused, she fails to define the processes necessary to identify scalability bottlenecks—failure number two. Because no one is identifying bottlenecks or chokepoints in the architecture, the user count or transaction volume exceeds a certain threshold and the entire product fails—failure number three. The team assembles to solve the problem, but because it has never invested in processes to troubleshoot incidents and their related problems, the team misdiagnoses the failure as “the database needs to be tuned”—failure number four. The vicious cycle goes on for days, with people focusing on different pieces of the technology stack and blaming everything from

firewalls, to applications, to the persistence tiers to which the apps speak. Team interactions devolve into shouting matches and finger-pointing sessions, while services remain slow and unresponsive. Customers walk away, team morale flat-lines, and shareholders are left holding the bag.

The key point here is that crises resulting from an inability to scale to end-user demands are almost never technology problems alone. In our experience as former executives and advisors to our clients, scalability issues start with organizations and people, and only then spread to process and technology. People, being human, make ill-informed or poor choices regarding technical implementations, which in turn sometimes manifest themselves as a failure of a technology platform to scale. People also ignore the development of processes that might help them learn from past mistakes and sometimes put overly burdensome processes in place, which in turn might force the organization to make poor decisions or make decisions too late to be effective. A lack of attention to the people and processes that create and support technical decision making can lead to a vicious cycle of bad technical decisions, as depicted in the left side of Figure I.1. This book is the first of its kind focused on creating a virtuous cycle of people and process scalability to support better, faster, and more scalable technology decisions, as depicted in the right side of Figure I.1.

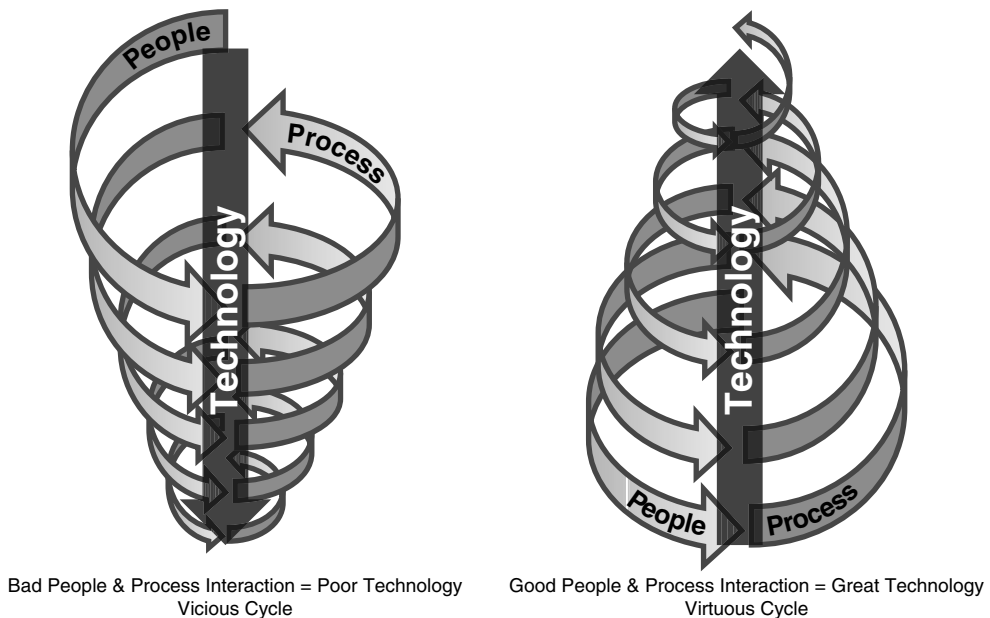


Figure I.1 *Vicious and Virtuous Technology Cycles Utility*

Art Versus Science

Our choice of the word *art* in the title of this book is a deliberate one. *Art* conjures up images of a fluid nature, whereas *science* seems much more structured and static. It is this image that we heavily rely on, as our experience has taught us that there is no single approach or way to guarantee an appropriate level of scale within a platform, organization, or process. A successful approach to scaling must be crafted around the ecosystem created by the intersection of the current technology platform, the characteristics of the organization, and the maturity and appropriateness of the existing processes. This book focuses on providing skills and teaching approaches that, if employed properly, will help solve nearly any scalability or availability problem.

This is not to say that we don't advocate the application of the scientific method in nearly any approach, because we absolutely do. *Art* here is a nod to the notion that you simply cannot take a "one size fits all" approach to any potential system and expect to meet with success.

Who Needs Scalability?

Any company that continues to grow ultimately will need to figure out how to scale its systems, organizations, and processes. Although we focus on Web-centric products through much of this book, we do so only because the most unprecedented growth has been experienced by Internet companies such as Google, Yahoo, eBay, Amazon, Facebook, LinkedIn, and the like. Nevertheless, many other companies experienced problems resulting from an inability to scale to new demands (a lack of scalability) long before the Internet came of age. Scale issues have governed the growth of companies from airlines and defense contractors to banks and colocation facility (data center) providers. We guarantee that scalability was on the mind of every bank manager during the consolidation that occurred after the collapse of the banking industry.

The models and approaches that we present in our book are industry agnostic. They have been developed, tested, and proven successful in some of the fastest-growing companies of our time; they work not only in front-end customer-facing transaction-processing systems, but also in back-end business intelligence, enterprise resource planning, and customer relationship management systems. They don't discriminate by activity, but rather help to guide the thought process on how to separate systems, organizations, and processes to meet the objective of becoming highly scalable and reaching a level of scale that allows the business to operate without concerns about its ability to meet customer or end-user demands.

Book Organization and Structure

We've divided the book into five parts. Part I, "Staffing a Scalable Organization," focuses on organization, management, and leadership. Far too often, managers and leaders are promoted based on their talents within their area of expertise. Engineering leaders and managers, for example, are very often promoted based on their technical acumen and aren't given the time or resources needed to develop their business, management, and leadership acumen. Although they might perform well in the architectural and technical aspects of scale, their expertise in organizational scale needs is often shallow or nonexistent. Our intent is to provide these managers and leaders with a foundation from which they can grow and prosper as managers and leaders.

Part II, "Building Processes for Scale," focuses on the processes that help hyper-growth companies scale their technical platforms. We cover topics ranging from technical issue resolution to crisis management. We also discuss processes meant for governing architectural decisions and principles to help companies scale their platforms.

Part III, "Architecting Scalable Solutions," focuses on the technical and architectural aspects of scale. We introduce proprietary models developed within AKF Partners, our consulting and advisory practice. These models are intended to help organizations think through their scalability needs and alternatives.

Part IV, "Solving Other Issues and Challenges," discusses emerging technologies such as grid computing and cloud computing. We also address some unique problems within hyper-growth companies such as the immense growth and cost of data as well as issues to consider when planning data centers and evolving monitoring strategies to be closer to customers.

Part V, "Appendices," explains how to calculate some of the most common scalability numbers. Its coverage includes the calculation of availability, capacity planning, and load and performance.

The lessons in this book have not been designed in the laboratory, nor are they based on unapplied theory. Rather, these lessons have been designed and implemented by engineers, technology leaders, and organizations through years of struggling to keep their dreams, businesses, and systems afloat. The authors have had the great fortune to be a small part of many of these teams in many different roles—sometimes as active participants, at other times as observers. We have seen how putting these lessons into practice has yielded success—and how the unwillingness or inability to do so has led to failure. This book aims to teach you these lessons and put you and your team on the road to success. We believe the lessons here are valuable for everyone from engineering staffs to product staffs, including every level from the individual contributor to the CEO.

This page intentionally left blank

This page intentionally left blank

Index

A

- About this book, 5, 9–10
- “Above the Clouds” (UC Berkeley), 475–476
- Abrams, Jonathan, 71
- ACA (Affordable Care Act), 331–332
- Accountability of CEO, 24–25
- ACID database properties, 400, 401, 441
- Acronyms
 - DRIER, 148–149, 157, 158
 - RASCI, 36–39, 40, 156, 214, 223
 - SMART goals, 89–90, 96, 97, 156
- AdSense, 91
- AdSonar, 91, 92
- Affective conflict
 - about, 54–55
 - defined, 13
 - influencing innovation, 63
- Affordable Care Act (ACA), 331–332
- Agile Organizations
 - about, 66–68, 70, 240–241, 247
 - aligning to architecture, 62
 - ARB process in, 246–247
 - autonomy of teams in, 241–242, 247
 - evolution of, 59–61
 - illustrated, 61
 - incorporating barrier conditions in, 293–295
 - limited resources in, 242–243, 247
 - maintaining standards across teams, 243–246, 248
 - scaling processes for, 16
 - team ownership of architecture, 241–242
 - tradeoffs in, 307–308, 312, 313
- AKF
 - AKF
 - AKF Application Scale Cube
 - implementing, 357–358
 - summary of, 365–367
 - using, 367–371
 - x*-axis, 357–358, 359, 361, 365–367, 371–372
 - y*-axis, 357–358, 361–362, 371, 372–373
 - z*-axis, 357–358, 359, 361–362, 363–364, 365–367, 371–373
 - AKF Database Scale Cube
 - applying, 375–376
 - business-to-business SaaS solutions with, 391–392
 - concerns about replication delays, 377
 - ecommerce implementation for, 388–389
 - employing for search implementation, 389–391
 - illustrated, 376
 - summary of, 385–388
 - timeline for employing splits, 393, 394
 - when and how to use, 392
 - x*-axis, 376–381, 386, 387, 393
 - y*-axis, 381–383, 386, 387, 388, 393–394
 - z*-axis, 383–386, 387, 388, 394
 - AKF Scale Cube. *See also* AKF Application Scale Cube; AKF Database Scale Cube
 - application state and, 351
 - with cloud environments, 485–487, 494
 - data segmentation scenarios for, 434
 - defined, 343–345
 - illustrated, 344, 350
 - implementing, 357–358
 - overview, 353–355
 - summary of, 350–352
 - uses for, 355
 - using for database splits, 375–376
- AKF
 - architectural principles of, 214–222
 - definition of management, 101
 - 5-95 Rule for, 104, 105, 117
 - risk model, 260

- AKF Scale Cube (*continued*)
 - when and where to use, 352–353
 - x-axis of, 344–346, 351, 352, 354
 - y-axis of, 346–348, 351, 352, 354
 - z-axis of, 349–350, 351, 352, 354
- Allchin, Jim, 43
- Allen, Paul, 256
- Allspaw, John, 159–160, 239
- Amazon, 16–17, 44, 123, 126, 281, 461, 462
- Amazon Web Services, 331, 488, 492, 516
- Amdahl's law, 448–449
- Amelio, Gil, 11, 257
- Apple Computer, 11, 257–258
- Application caches, 401
- Application servers
 - capacity planning calculations for, 552
 - monitoring requests for, 548–549
- Application service providers (ASPs), 461
- Applications. *See also* Preparing cloud applications; Splitting applications
 - AKF Scale Cube and, 351
 - cache hits/misses for, 397, 409
 - caching software, 406–407
 - calculating server capacities for, 552
 - designing for monitoring, 495, 507
 - grid computing and monolithic, 452–453, 458
 - portability between clouds, 472, 474
 - rapid development of, 296–297
 - stateful and stateless, 218–219, 418–419
 - user sessions for stateful, 420–422, 424
 - y-axis splits for complexity and growth in, 371
- Arao, Karl, 206
- ARB (Architecture Review Board)
 - as barrier condition, 292, 294, 296, 302
 - checklist for, 236
 - considering members of, 230–232
 - defined, 225
 - entry/exit criteria for, 234–235
 - feature approval by, 230
 - following TAD rules, 324
 - implementing JAD and, 237
 - meetings of, 232–234
 - overview, 237, 238
 - process in Agile Organizations, 246–247
 - using AKF Scale Cube, 353
- Architects, 30–31
- Architectural principles. *See also* Technology-agnostic design
 - asynchronous design, 217–218, 222
 - automation over people, 221–222
 - building small, releasing small, failing fast, 221, 222
 - buy when non-core, 220, 222
 - D-I-D matrix, 306–307
 - design for disabling systems, 215, 222, 300–301, 302
 - designing for rollback, 215, 222, 302
 - developing, 209–211, 223
 - engendering ownership of, 213–214
 - following RASCI principle, 214, 223
 - illustrated, 213
 - isolating faults, 221, 222
 - mature technologies, 217, 222
 - monitoring as, 215–216, 222
 - most adopted AKF, 214–222
 - multiple live sites, 216, 222
 - N + 1 design, 213–215, 222
 - providing two axes of scale, 219–220, 222
 - scaling out, not up, 219, 222
 - scope of, 223
 - selecting, 212–213
 - SMART characteristics of, 211, 223
 - stateless systems, 218–219, 222
 - team development of, 223
 - using commodity hardware, 220, 222
- Architecture. *See also* Fault isolation
 - aligning Agile Organizations to, 62
 - designing for any technology, 317–318
 - fault isolation terminology, 327–329
 - implementing fault isolation, 339–341
 - multiple live data centers, 527
 - object cache, 401
 - OSI Model, 460
 - ownership by Agile teams, 241–242
 - technology-agnostic, 318–319, 323–325
- Architecture Review Board. *See* ARB
- Art of Capacity Planning, The* (Allspaw), 159
- Art of scalability, 4, 531–532
- Artificial intelligence, 461
- ASPs (application service providers), 461

- Asynchronous design
 - as architectural principle, 217–218, 222
 - coordination and communication in, 415
 - data syncing methods for, 411–412, 423
 - HTTP's stateless protocol, 418, 424
 - initiating new threads, 413, 423
 - scaling synchronously or asynchronously, 414–415
 - synchronous vs. asynchronous calls, 412–413
 - systems using, 416–418
 - ATM (Asynchronous Transfer Mode) networks, 460
 - Atomicity of databases, 401
 - Automating processes, 221–222
 - Autonomic Computing Manifesto (IBM), 461, 482
 - Availability
 - calculating Web site, 539
 - customer complaints as metric for, 541–542
 - determining with third-party monitoring services, 543–544
 - fault isolation and, 329–333, 334, 342
 - graphing calculations of, 544–545
 - guaranteeing transaction, 378
 - increasing with fault isolation, 334, 342
 - measuring, 112
 - monitoring portion of site down, 542–543
 - TAD and, 323, 326
 - Avoidance in user sessions, 420–421, 422
 - Axes of scale, 219–220, 222
- B**
- Back-office grids, 456–457
 - Bank of America, 320
 - Barrier conditions
 - ARB process as, 292, 294, 302
 - creating for hybrid development models, 296–297
 - establishing performance testing for, 292
 - including in Agile development, 293–295
 - JAD process as, 294
 - overview, 301, 302
 - uses for, 291–293
 - waterfall development and, 295–296
 - Batch cache refresh, 397
 - Behavior
 - evaluating employee, 109–110
 - leaders and selfless, 79–80
 - leadership influencing, 96
 - Bezos, Jeff, 16–17
 - Blackouts, 112
 - Blah* as a Service offerings, 461–462
 - Blink* (Gladwell), 262
 - Blogs on scalability, 535
 - “Blue-eyed/brown-eyed” exercise, 54, 55–56
 - Board of directors, 37
 - Books on scalability, 535
 - Boundaries
 - fault isolation and swim lane, 338, 342
 - finding optimal team, 44
 - team, 65
 - Boyatzis, Richard, 76–77
 - Brewer, Eric, 378
 - Brewer's theorem, 378
 - Brooks, Jr., F. P., 14, 16, 46, 448
 - Brownouts, 112
 - Budgets for headroom, 198, 208
 - Buffers vs. caches, 396, 409
 - Build grids, 455–456
 - Build small, release small, fail fast, 221, 222
 - Build vs. buy decisions
 - checklist of questions for, 255, 258, 321–322
 - considering strategic competitive differentiation, 253, 255
 - cost-effectiveness of component building, 254–255
 - cost-focused approaches to, 250–251
 - developing and maintaining components in, 253–254, 255
 - estimating competition for component, 254, 255
 - failures in build-it-yourself decisions, 256–258
 - making good buy decisions, 255–256
 - merging cost and strategy approaches in, 252–253, 258

- Build vs. buy decisions (*continued*)
 - Not Built Here phenomenon and, 252
 - overview, 258
 - scalability and, 249–250
 - strategy-focused approaches to, 251, 258
 - TAD and, 323, 325
- Bureaucracy, 140
- Business change calendar, 187
- Business unit owners, 27–28
- Buy when non-core, 220, 222

- C
- Cabrera, Felipe, 492
- Caches
 - buffers vs., 396, 409
 - cache hits, 397
 - cache misses, 397, 398
 - cache ratio, 397
 - object, 399–402
 - proxy, 402–403
 - refreshing batch, 397
 - reverse proxy, 403–405
 - types of application, 401
- Caching
 - content delivery networks, 407–408
 - defined, 395–396, 409
 - HTML headers vs. meta tags for
 - controlling, 405
 - LRU algorithm for, 397–398, 408
 - MRU algorithm for, 398, 408
 - software, 406–407
 - structures for, 396, 409
- Calculating
 - hardware uptime, 540–541
 - headroom, 205, 206, 200–201
 - headroom capacity, 547–553
 - load and performance, 555–561
 - load and performance for SaaS, 555–561
 - tradeoffs using decision matrix, 309
 - Web site availability, 539–545
- Callbacks, 415
- CAP theorem, 378
- Capability levels, 134–135
- Capability Maturity Model Integration (CMMI) project, 132, 134–135, 136–137, 140
- Capacity
 - calculating headroom, 547–553
 - maximizing with grid computing, 450, 451
 - planning, 33–34, 203–205, 208
- Carnegie Mellon Software Engineering Institute, 134
- Case methods
 - about, 9–10
 - airline pricing models, 368–370
 - Amazon, 16–17, 44, 123, 126, 281, 461, 462
 - Amazon Web Services, 331, 488, 492, 516
 - Apple, 11, 257–258
 - eBay, 34–35, 123, 162–163, 388–389
 - Etsy, 159–160, 239–240
 - FAA's Air Traffic Control system, 181
 - Friendster, 71–72
 - Google, 91, 123, 462
 - Google MapReduce, 435–438, 444, 457
 - Intuit, 99–100, 102–105, 491–493
 - Microsoft, 43, 99, 256, 462
 - Netflix, 281
 - PayPal, 123
 - Quigo, 91, 92–94, 114–115, 210
 - Salesforce, 330–331, 391–392
 - Spotify, 68–69, 244–245
 - Wooga, 244, 245–246
- Causal roadmap to success, 94–95, 97
- CD. *See* Continuous delivery systems
- CDNs (content delivery networks), 407–408, 409–410
- Centralization in user sessions, 421, 422
- CEOs (chief executive officers). *See also* Leaders
 - accountability of, 24–25
 - RASCI matrix and, 36–38
 - role of, 25–26, 40
- CFOs (chief financial officers), 27–28
- Change control
 - change control meetings, 191–192, 195
 - performance/stress testing and, 288
- Change identification
 - about, 179–180, 194
 - change management vs., 181
- Change log, 179

- Change management
 - about, 180–183, 193–195
 - approving changes, 186–187, 194
 - change control meetings, 191–192, 195
 - checklist for, 193
 - continuous process improvement, 192, 195
 - FAA’s Air Traffic Control system, 181
 - identifying change, 179–180, 181, 194
 - implementing and logging changes, 189, 194
 - ITIL goals of, 183
 - proposing changes, 183–186, 194
 - reviewing changes, 191, 194, 195
 - rollback plans for, 190
 - scheduling changes, 187–189, 194
 - validating changes, 189–190, 194
- Changes. *See also* Change management
 - about, 177–178
 - approving, 186–187, 194
 - defining, 178
 - implementing and logging, 189, 194
 - postmortems for, 153–156, 158, 172–173, 175
 - proposing, 183–186, 194
 - reviewing, 191, 194, 195
 - rollback plans for, 190
 - scheduling, 187–189, 194
 - using crises as catalyst for, 162–163
 - validating, 189–190, 194
- Chaos in crisis, 163–164, 174
- Chapters, 69, 245
- Chat channel, 166, 168–169, 175
- Checklists
 - Architecture Review Board review, 236
 - build vs. buy questions, 255, 258, 321–322
 - change management, 193
 - fast or right, 310–311
 - fault isolation design, 340–341
 - headroom calculation, 205
 - joint architecture design sessions, 227–228, 236
 - markdown, 301
 - performance testing steps, 280
 - risk assessment steps, 268
 - rollbacks, 298
 - team size, 50–51
- Chief executive officers. *See* CEOs
- Chief financial officers (CFOs), 27–28
- Chief technology officers. *See* CTOs
- Chipsoft, 99
- Chunk, 342
- CIOs (chief information officers). *See* CTOs
- Citibank, 320
- Cloud, 460
- Cloud computing. *See also* IaaS; Preparing cloud applications
 - application portability in, 472, 474
 - benefits of, 468–471, 481
 - Blah* as a Service offerings, 461–462
 - common characteristics of, 466
 - control issues in, 472–473, 475
 - cost of, 469–470, 471, 474, 475
 - decision making steps for, 478–481, 482, 483
 - decision matrix, 479–480
 - drawbacks of, 471–476, 482–483
 - fitting to infrastructures, 476–478, 482, 483
 - flexibility of, 470–471
 - grids vs., 467–468
 - history of, 460–461, 481, 482
 - multiple tenants, 465
 - overview, 459, 481–483
 - pay by usage for, 463
 - performance of, 473–474, 475
 - public vs. private clouds, 462–463
 - scale on demand, 464–465
 - security liabilities of, 471–472, 474
 - skill sets needed for, 478
 - speed benefits of, 470, 471
 - UC Berkeley’s assessment of, 475–476
 - using with production environments, 476–478
 - virtualization, 466–467
- Clusters, 328, 329
- COBIT (Control Objectives for Information and Related Technology), 143, 144
- Code reviews
 - introducing, 292
 - using with RAD method, 296
 - waterfall development and, 296

- Coding. *See* Source code
- Cognitive conflict
about, 54–55
defined, 13
influencing innovation, 63
- Collins, Jim, 79
- Communications
during crises and escalations, 168–169, 171–173, 175
effects of experience on, 119–120
making customer apologies, 173
organizational influences in, 41–42
team size and poor, 47
within functional organizations, 53–54
- Companies. *See* Organizations
- Complexity
data splitting for growth and, 371, 383
grid computing, 453, 458
of processes, 137–139
- Components
considering scalability and, 321–322
cost-effectiveness of building, 254–255
developing and maintaining assets or, 253–254, 255
estimating competition for, 254, 255
example of good buy decisions, 255–256
failures in build-it-yourself decisions, 256–258
strategic competitive differentiation for, 253, 255
- Computers. *See also* Servers
decreasing size of, 509–510
processing large data sets with
distributed, 434–438, 444
- Conflicts
“blue-eyed/brown-eyed” exercise, 54, 55–56
cognitive or affective, 54–55
incident and problem resolution, 150
types of, 13–14
when process not fitted to culture, 139–140, 141, 142
within organizations, 54, 66, 67
- Consistency
database, 401
node, 378
- Content delivery networks (CDNs), 407–408, 409–410
- Continuous delivery (CD) systems
about, 182
change approvals in, 187
change control meetings for, 192
unit testing in, 293
- Continuous process improvement, 192, 195
- Control in cloud computing, 472–473, 475
- Control Objectives for Information and Related Technology (COBIT), 143, 144
- Cook, Scott, 99
- Costs. *See also* Tradeoffs
benefits using grid computing, 451
cloud computing, 469–470, 471, 474, 475
cost-value data dilemma, 430–431
data center, 509–511, 520, 521–525
data storage, 427–429, 432–433, 444
factoring in project triangle, 303–306, 313
focus in build vs. buy decisions, 250–251, 258
measuring cost of scale, 111–112
projecting data center, 515
reducing with fault isolation, 335–336
rollback, 299–300
technology-agnostic design and, 319–320, 326
y-axis splits, 348
- Cowboy coding, 295
- Creativity, 133–134
- Crises and escalations
about escalations, 170–171, 175
characteristics of crises, 160–161
communications and control in, 168–169, 175
crises vs. incidents, 161–162, 174
eBay scalability crisis, 162–163
engineering lead’s role in, 167–168, 174
Etsy’s approach to, 159–160
from incident to crisis, 161
individual contributor’s role in, 167–168, 174
managing, 163–168
postmortems and communications
about, 172–173, 175
problem manager’s role, 164–166, 174
status communications in, 171–172

- team manager's role, 166–167
 - using as catalyst for change, 162–163
 - war rooms for, 169–170, 175
 - Crisis managers, 168, 174
 - Crisis threshold, 161
 - CTOs (chief technology officers)
 - experiential chasm with, 121–122, 128
 - problems with, 121–122
 - RASCI matrix and, 37, 38
 - responsibility for scalability, 2–3
 - role of, 28–30, 40
 - Cultures
 - candidates' fit into, 108, 116
 - clashes with processes, 139–140, 141, 142
 - productivity and behavior in, 11
 - Customers
 - apologizing to, 173
 - handling growing customer base, 371
 - impact of downtime on, 543
 - unprofitable, 431
 - using complaints as metric, 541–542
- D**
- D-I-D matrix, 307
 - Daily incident meetings, 152–153, 157, 158
 - Data. *See also* Data storage; Source code; Splitting databases
 - analyzing performance test, 278–279, 280, 555, 559–561
 - assessing stress test, 285, 286
 - caching, 395–399
 - collecting repeat test, 279–280
 - collecting to identify problems, 498–499
 - cost-value dilemma for storing, 430–431
 - costs of, 427–429, 444
 - ETL concept for, 434, 439, 457
 - high computational rates of grid
 - computing, 450, 451, 458
 - methods for synching, 411–412, 423
 - NoSQL solutions for scalability, 440–443, 445
 - plotting on control charts, 560
 - processing large data sets, 434–438, 444
 - separating by meaning, function, or usage, 381–383
 - transforming, 433–434
 - y- and z-axis Big Data splits, 438–440, 445
 - Data centers
 - considering multiple, 525–527
 - costs of, 509–511, 520, 521–525
 - IaaS strategies vs., 516–519
 - location of, 511–514
 - projecting growth for, 514–516
 - Rules of Three for, 519–525
 - splitting, 521–525
 - Data sets
 - processing large, 434–438, 444
 - using y- and z-axis Big Data splits, 438–440, 445
 - Data storage
 - cost-value dilemma for, 430–431
 - costs of, 427–429, 432–433, 444
 - issues in, 427
 - matching value to costs of, 431–434
 - option value of data, 431–432
 - overview, 444–445
 - reducing data set size, 434–438, 444
 - strategic competitive differentiation, 432
 - tiered storage solutions, 432–433, 444
 - transforming data, 433–434
 - types of costs in, 429
 - Data warehouse grids, 456
 - Database servers, 548–549, 552
 - Databases. *See also* Splitting databases
 - ACID properties of, 400, 401, 441
 - calculating capacity for, 552
 - cloning data without bias, 375, 376–377
 - replication delays in, 377, 380
 - using axes splits for, 387–388
 - Datum, 396, 399, 409
 - DDOS (distributed denial-of-service)
 - attacks, 490
 - Deadlock, 412
 - Decentralized user sessions, 421, 422
 - Decision making. *See also* Build vs. buy decisions
 - analyzing tradeoffs, 303–313
 - ARB, 233–234
 - evaluating management's, 106
 - leadership and, 81–82
 - mistakes in, 81
 - snap judgements and, 262
 - steps for cloud computing, 478–481
 - Decision matrix
 - calculating tradeoffs with, 308, 309, 313
 - for cloud computing, 479–480

- Delegation
 - by CEO, 26
 - by CTO/CIO, 29
 - guidelines for, 24–25
 - Destructive interference, 120
 - Development life cycle, 302
 - DevOps responsibilities, 31–32
 - Disabling systems
 - architectural design for, 215, 222, 300–301, 302
 - markdown checklist for, 301
 - Distributed denial-of-service (DDoS) attacks, 490
 - Diversity
 - experiential, 63–64
 - network, 64, 242
 - Documentation, 230, 234, 235
 - DRIER process, 148–149, 157, 158
 - Dunning, David, 76
 - Dunning-Kruger effect, 76, 77
 - Durability of databases, 401
- E**
- eBay, 34–35, 123, 162–163, 388–389
 - Ecommerce scalability, 388–389
 - Economies of scale, 244, 248
 - Efficiency in organizations, 41–44
 - 80/20 rule, 276
 - Elliott, Jane, 55–56
 - Employees
 - cultural and behavioral fit of, 108
 - recruiting for data centers, 514
 - signs of under- and overworked, 47–48
 - team size and experience of, 45
 - terminating, 109–110
 - Empowerment
 - JAD entry/exit criteria for team, 229
 - team, 64–65
 - Engineers
 - cowboy coding by, 295
 - escalating crises to managers, 171
 - fostering technology agnosticism in, 325
 - individual contributions by, 31
 - infrastructure, 32–33
 - measuring productivity of, 113–114
 - reporting performance test results to, 279, 280, 556, 561
 - role in crises, 167–168, 174
 - Entry/exit criteria
 - joint architecture design, 228–230
 - used for waterfall implementations, 295–296
 - Environments. *See* Preparing cloud applications; Production environments
 - Equations for headroom, 201, 202, 551
 - Escalations. *See* Crises and escalations
 - Ethics
 - leadership and, 78–79, 96
 - managerial, 100–101
 - ETL (extract, transform, and load) concept, 434, 439, 457
 - Etsy, 159–160, 239–240
 - Everything as a Service (XaaS), 461, 462, 483
 - Executing performance tests, 277–278, 280, 555, 557–559
 - Executive interrogation, 26
 - Executives
 - business unit owners, general managers, and P&L owners, 27–28
 - CEOs, 25–26, 40
 - CFOs, 27–28
 - CTO/CIO, 28–30
 - experiential chasm with, 120–121, 128
 - Experiential chasm, 119–120, 128
 - Experiential diversity, 63–64
 - Extract, transform, and load (ETL) concept, 434, 439, 457
- F**
- FAA's Air Traffic Control system, 181
 - Facebook, 71, 72
 - Fail Whale, 197
 - Failure domains, 49, 50, 147
 - Failure mode and effects analysis (FMEA), 264–267, 268, 270–271
 - Failures
 - build small, release small, fail fast design principle, 221, 222
 - cloud environment outages, 487–489, 494
 - communication, 41–42
 - effects of, 21–23
 - leadership, 71–72
 - Microsoft's Longhorn, 43

- Fannie Mae, 320
- Fast or right checklist, 310–311
- Fault isolation
 - along swim lane boundary, 338, 342
 - approaches to, 336–367
 - architectural terms for, 327–329
 - benefits of, 329–336
 - challenges for cloud applications, 487–489, 494
 - costs and, 335–336
 - design checklist for, 340–341
 - examples of, 327
 - implementing, 339–341
 - increasing availability with, 329–333, 334, 342
 - no shared components or data, 337
 - scalability and, 334
 - testing designs for, 341
 - time to market and, 334–335
 - with transactions along swim lanes, 338
- Fault isolation zones, 221, 222, 338
- Features
 - analyzing tradeoffs for, 307–310
 - documenting tradeoffs for, 230, 234, 235
 - JAD entry/exit criteria for, 228–230
 - requiring approval by ARB, 230
- Federal Aviation Administration (FAA) Air Traffic Control system, 181
- FeedPoint, 91
- 5-95 Rule, 104, 105, 117
- Flexibility
 - cloud computing, 470–471
 - NoSQL solution and query, 442
- FMEA (failure mode and effects analysis), 264–267, 268, 270–271
- Ford, Henry, 347
- Forward proxy cache, 402–403
- Foster, Ian, 447
- Freddie Mac, 320
- Friendster, 71–72
- Functional organizational structure
 - characteristics of, 51–56, 70
 - communications within, 53–54
 - conflicts within, 54, 66, 67
 - illustrated, 52
 - matrix organizations vs., 58
- G
 - Galai, Yaron, 91
 - Gates, Bill, 26, 43, 256
 - Gateway caches, 404, 409
 - General managers, 27–28
 - Gladwell, Malcolm, 262
 - Globus Toolkit, 447, 448
 - Go/no-go processes. *See* Barrier conditions
 - Goal trees, 114–115, 118, 209–210
 - Goals
 - applying to scalability solutions, 93–94
 - change management, 183
 - creating goal tree, 114–115
 - D-I-D matrix for project, 307
 - defined, 89, 97
 - developing architectural principles from, 209–211, 223
 - ineffectiveness in shared, 22–23
 - SMART, 89–90, 96, 97, 156, 211
 - Good to Great* (Collins), 79
 - Google, 91, 123, 462
 - Google MapReduce, 435–438, 444, 457
 - Graph databases, 443
 - Graphs
 - headroom, 207
 - Web site availability, 544–545
 - Grid computing
 - back-office grids, 456–457
 - build grids in, 455–456
 - cloud computing vs., 467–468
 - complexity of, 453, 458
 - cons of, 452–453
 - data warehouse grids, 456
 - high computational rates of, 450, 451, 458
 - history of, 447–449
 - implemented in MapReduce, 457
 - maximizing capacity used, 450, 451
 - monolithic applications and, 452–453, 458
 - overview, 457–458
 - production grids in, 454
 - pros of, 449–451, 458
 - shared infrastructure for, 450, 451, 452, 458
 - Grid, The* (Foster and Kesselman), 447

Growth

- dealing with too much data, 427
 - own/lease/rent options plotted against, 518–519
 - projecting data center, 514–516
 - projecting in headroom calculations, 200–201
 - using AKF Database Scale Cube for, 392
 - y-axis application splits for, 371
 - y-axis database splits for, 383
 - z-axis database splits for, 383–385
 - z-axis splits for customer base, 371
- Guilds, 69, 245
- Gut-feeling risk assessment, 261–263, 271, 308, 313

H

- Half racks, 510
- Hardware, 220, 222
- Headroom
- calculating, 205
 - determining, 199–203
 - equations for, 201, 202, 551
 - ideal usage percentages, 203–205
 - overview, 207–208
 - performance/stress testing and, 288
 - planning, 198–199
 - spreadsheet for, 206–207
- Healthcare.gov, 331–332, 333
- Heat maps, 498
- Hewlett-Packard, 462
- Hiring
- cultural interviews before, 108, 116
 - headroom calculations and, 198, 208
 - selecting candidates, 107–108
- Hit ratio, 397, 409
- Hoare, Sir Charles Anthony Richard, 412
- Homo homini lupus* strategies, 65
- Hotlinks, 71
- HTTP (Hyper-Text Transfer Protocol)
- headers in, 405
 - stateless protocol for, 418, 424
- Human factors in risk management, 270–271
- HVAC services, 510, 512, 513, 528, 529

I

- IaaS (Infrastructure as a Service)
- concept of, 461, 462, 482
 - PaaS vs., 489
 - scaling features for, 219
 - shifting from data centers to, 516–519, 527–528
- IBM, 256, 461, 482
- Incident monitors, 500–501
- Incidents. *See also* Crises and escalations
- assessing data for, 503–504
 - assigning swim lanes to sources of, 339–340
 - conflicts in managing, 150
 - crises vs., 161–162, 174
 - daily meetings about, 152–153, 157, 158
 - defined, 144–145, 158
 - DRIER process for managing, 148–149, 157, 158
 - escalating to crisis, 160–161
 - finding, 496–503, 507
 - life cycles for, 150–151
 - management components of, 146–149
 - monitoring with failure domains, 147
 - overlooking, 495–496
 - postmortems for, 153–156, 158
 - quarterly reviews of, 153, 157, 158
 - resolving, 147
- Individual contributors
- architects, 30–31
 - capacity planning, 33–34
 - DevOps responsibilities, 31–32
 - engineers, 31
 - infrastructure engineers, 32–33
 - quality assurance, 33
- Information Technology Infrastructure Library (ITIL), 143, 144, 145, 146, 148, 149, 183
- Infrastructure as a Service. *See* IaaS
- Infrastructure engineers, 32–33
- Infrastructures. *See also* IaaS
- fitting cloud computing to, 476–478, 482, 483
 - sharing for grid computing, 450, 451, 452, 458
- Initiating new threads, 413, 423

Innovation

- cognitive/affective conflict and, 63, 66
- defined, 62–63
- experiential diversity and, 63–64
- network diversity, 64, 242
- sense of empowerment and, 64–65, 66
- team boundaries and, 63
- theory of innovation model, 66

Input/output per second (IOPS), 489, 494

Internet pipe costs, 512

Intuit, 99–100, 102–105, 491–493

IOPS (input/output per second), 489, 494

IRC channels, 166, 168–169, 175

Isaacson, Walter, 11, 257–258

Isolation of databases, 401

Issue management, defined, 144

IT organization model, 122–124

ITIL (Information Technology Infrastructure Library), 143, 144, 145, 146, 148, 149, 183

ITSM (IT Service Management) framework, 183

Itzhak, Oded, 91

Ivarsson, Anders, 68, 244

J

JAD (joint architecture design)

- checklist for, 227–228, 236
- defined, 225
- designing for Agile teams, 242, 247
- entry/exit criteria for, 228–230
- fixing organizational dysfunction with, 225–226
- following TAD rules, 324
- function of, 226–227
- implementing ARB and, 237
- membership of, 237
- overview, 236–237
- using AKF Scale Cube with, 353
- using as barrier condition, 294

JavaScript Object Notation (JSON), 443

Jobs, Steve, 10–11, 257–258

Joint architecture design. *See* JAD

K

Keeven, Tom, 34–35, 330

Kesselman, Carl, 447

Key performance indicators (KPIs), 496

King, Jr., Martin Luther, 54, 55

KLOC (thousands of lines of code), 113–114

Kniberg, Henrik, 68, 244

KPIs (key performance indicators), 496

Kruger, Justin, 76

L

Law of the Instrument, The, 34

Leaders

- aligning to shareholder value, 83, 96
 - asking questions, 25–26
 - behavior of, 11
 - born or made, 73, 96
 - building team relationships, 119–122, 128
 - dealing with conflicts, 55
 - decision making by, 81–82
 - delegation by, 24–25
 - developing causal roadmap to success, 94–95, 97
 - developing vision statements, 84–87
 - empowering teams and scalability, 82–83
 - implementing scalability, 532–534
 - making build vs. buy decisions, 249–258
 - morality of, 75, 81–82, 96
 - overview, 95–97
 - problems with executives, 120–121
 - resolving crises, 174
 - scalability proficiency of, 26
 - seeking outside help, 26
 - selfless behavior of, 79–80
 - setting examples, 78–79, 96
 - SMART goals developed by, 89–90
 - 360-degree reviews for, 77, 96
 - transformational leadership by, 84, 96
 - valuing people, 80–81
- Leadership. *See also* Leaders
- assessing abilities for, 76–78, 96
 - attributes of, 74–75, 96

- Leadership (*continued*)
- creating mission statements, 87–89
 - decision making and, 81–82
 - defined, 72–73, 96
 - developing qualities for, 73, 96
 - ethics and, 78–79, 96
 - failures in, 71–72
 - importance in scalability, 17–19, 20
 - management vs., 17–18, 73, 101
 - model of, 74–76
 - overview, 95–97
 - selfless behavior and, 79–80
 - transformational, 84, 96
 - working with limited Agile resources, 242–243, 247
- Life cycles
- development, 302
 - problem and incident, 150–151
- Live Community, 492–493
- Loads. *See also* Performance testing
- calculating performance of, 555–561
 - performance testing for server, 274
 - stress testing, 283, 284, 286
- Location of data centers, 511–514
- Lockheed Martin, 259
- LRU (least recently used) caching
- algorithm, 397–398, 408
- M**
- Management. *See also* Managers;
- Managing incidents and problems
 - building teams, 105–107
 - contingencies for project, 104–105
 - creating goal tree, 114–115
 - creating team success, 115–116, 117, 118
 - defining, 100–101, 116, 117
 - developing crisis management process, 163–164
 - ethics in, 100–101
 - evaluating measurement metrics and goals, 111–114, 117, 118
 - experiential chasm with teams, 119–120, 128
 - 5-95 Rule for, 104, 105, 117
 - implementing scalability, 532–534
 - importance in scalability, 17–19, 20
 - interviewing candidates, 108, 116
 - leadership vs., 17–18, 73, 101
 - leading postmortems, 153–156, 158
 - managing chaos in crisis, 163–164, 174
 - measuring output, 12–13
 - overview, 116–118
 - problem managers, 164–166, 174
 - problems with business leaders, 120–121
 - problems with CTOs, 121–122
 - project and task, 102–105
 - recognizing badly fitted processes, 139–140, 141, 142
 - similarities with leadership, 117
 - team managers in crises, 166–167
 - upgrading teams, 107–111
 - using IT model for customer product, 122–124
 - when to implement processes, 137
- Managers
- appointing, 49, 51, 52
 - creating team success, 115–116
 - crisis, 168, 174
 - determining crisis threshold, 161
 - good, 102
 - measuring performance, 111–114
 - problem, 164–166
 - responsibilities of, 45–46
 - seed, feed, and weed activities for, 107–111, 117
 - selecting employment candidates, 107–108
 - team size and experience of, 45
 - terminating employees, 109–110
 - working with limited Agile resources, 242–243, 247
- Managing incidents and problems
- about incidents, 144–145
 - components of incident management, 146–149
 - components of problem management, 149–150
 - conflicts when, 150
 - daily incident meetings, 152–153, 157, 158
 - defining problems, 145–146, 158
 - DRIER process, 148–149, 157, 158
 - flow for, 156–157

- incident and problem life cycles, 150–151
- monitoring systems for, 147
- overview, 143–144
- postmortems, 153–156, 158
- quarterly incident reviews, 153, 157, 158
- Manifesto for Agile Software Development*, 59, 293
- MapReduce, 435–438, 444, 457
- Markdown functionality, 300–301, 302, 333
- Marshalling processes, 399
- Maslow’s Hammer, 34
- Matrices
 - D-I-D, 307
 - decision, 308, 309
 - RASCI, 35–39
- Matrix organizations
 - characteristics of, 56–59, 70
 - functional organizations vs., 58
 - illustrated, 57
 - moving goal lines in, 66
- Mature technologies, 217, 222
- Maturity levels, 134–135, 140
- MBPS (megabytes per second), 489, 494
- McCarthy, John, 461
- McKee, Annie, 76–77
- Mealy machine, 419
- Measurements. *See also* Metrics; Risks
 - cost of scale, 111–112
 - evaluating metrics and goals for, 111–114, 117, 118
 - managers’ support for, 102
 - measuring availability, 124–126
 - using as barrier conditions, 293, 295
- Meetings
 - Architecture Review Board, 232–234
 - change control, 191–192, 195
 - daily incident, 152–153, 157, 158
- Megabytes per second (MBPS), 489, 494
- Membership
 - Architecture Review Board, 230–232
 - joint architecture design, 237
- Metrics
 - customer complaints used as, 541–542
 - deriving from performance testing, 274
 - finding incidents using, 499–501
 - measuring output with, 12–13
 - needed for project, 111–114, 117, 118
- Micromanagement, 48
- Microsoft, 43, 99, 256, 462
- Mission, 97
- Mission First, People Always, 80–81, 96
- Mission statements
 - applying to scalability solutions, 92–93
 - creating, 87–89, 97
- Mitigating failure, 265–267
- Monitoring
 - applications, 503
 - correlating data size to problem specificity, 498
 - designing systems for, 215–216, 222, 495–503, 507
 - establishing barrier conditions with, 293
 - existing platforms for, 503, 507
 - failure domains, 147
 - learning what to monitor, 496–499
 - overview, 506–507
 - processes, 504–507
 - stress test processes, 284, 286
 - user experience and business metrics for, 499–501
 - using system, 501–502
 - value of, 503–504
 - Web and application server requests, 548–549
- Monolithic applications, 452–453, 458
- Moore, Gordon, 509
- Moore machine, 419
- Moore’s law, 219, 509
- Morale, 22, 47
- Morality of leaders, 75, 81–82, 96
- MRU (most recently used) caching
 - algorithm, 398, 408
- Multiple live sites, 216, 222, 521–528, 529
- Multitenant states, 219
- Multitenants
 - cloud computing with, 465
 - using SaaS database splitting for products, 391–392
- Mutex synchronization, 411, 423
- Mythical Man-Month, The* (Brooks, Jr.), 14, 16, 46, 448

N

NBH (Not Built Here) phenomenon, 252
 Negative stress testing, 182
 Netflix Chaos Monkey, 281
 Network architecture, 460
 Network diversity, 64, 242
 NeXT, 11, 257, 258
 N + 1 design, 213–215, 222
 No shared components or data, 337
 NoSQL solutions
 implementing, 434–438, 440, 444
 scalability using, 440–443, 445
 using multiple nodes in, 440
 Not Built Here (NBH) phenomenon, 252

O

Object caches, 399–402
 Office of Government Commerce (United Kingdom), 143, 146
 Option value of data, 431–432
 Organizational cost of scale, 14–17
 Organizational design. *See also* Joint architecture design
 Agile Organizations, 59–70
 cognitive conflict and, 13–14
 creating efficiency with, 41–44
 determining team size, 44–51
 functional organizational structure, 51–56, 70
 matrix organizational structure, 56–59, 70
 overview, 69–70
 signs of incorrect team size, 47–48
 Organizations. *See also* Agile Organizations; Organizational design
 choosing application splits for, 370–371
 clarity of roles in, 21–23, 40
 costs and build vs. buy decisions, 250–251, 258
 creating mission statements, 87–89
 customer apologies by, 173
 defining team roles, 23–24
 delegation within, 24–25
 designing, 20
 fitting cloud computing to, 476–478, 482, 483

 fostering standards within, 42–43
 introducing processes into, 135–139
 JAD for dysfunctional, 225–226
 mapping goals for, 114–115
 planning scalability for, 532–534
 as scalability element, 11–17, 20
 scalability needs of, 4
 strategy-focused build vs. buy decisions, 251, 258
 vision statements for, 86–87
 OSI Model, 460
 Outage costs, 126–127, 128
 Own/lease/rent options, 517, 518–519
 Ownership
 assigning for team processes, 140, 142
 engendering team architectural, 213–214, 241–242
 product standards and, 44

P

P&L owners, 27–28
 PaaS (Platform as a Service), 462, 483, 489
 Paging, 490
 Pareto, Vilfredo Federico Damaso, 276
 Partition tolerance in distributed computer systems, 378
 Paterson, Tim, 256
 Patient Protection and Affordable Care Act (PPACA), 331–332
 Pay-as-you-go cloud computing, 463
 PayPal, 123
 People
 assessing abilities of, 76–78, 96
 conflict between groups of, 54
 importance in scalability, 10–11, 20, 61, 531–532
 leader's valuing of, 80–81
 leadership attributes in, 74–75, 96
 managing, 116–118
 team size and productivity of, 15
 Performance. *See also* Caching; Performance testing
 buffers and caches for, 409
 cloud computing, 473–474, 475
 data thrashing, 203–204

- identifying stress test objectives, 281–282, 285
- management measurement of, 111–114, 118
- metrics and goals for, 111–114, 117, 118
- variability in cloud I/O, 489–491, 494
- Performance testing
 - analyzing data from, 278–279, 280, 555, 559–561
 - appropriate environments for, 275–276, 280, 289, 555, 556
 - defining, 276–277, 289, 555, 556
 - executing, 277–278, 280, 555, 557–559
 - goals of, 289
 - load testing, 274, 289
 - overview, 288–290
 - performing, 273–274
 - relating to scalability, 287–288, 290
 - repeating and analyzing, 279–280, 556, 561
 - reporting results of, 279, 280, 556, 561
 - steps in, 280, 289, 555–561
 - success criteria for, 274, 280, 555, 556
 - using as barrier condition, 292
- Pixar, 11
- Planning
 - capacity, 33–34
 - headroom, 198–199, 203–205, 208
 - organization’s scalability, 532–534
 - performance test definition, 276–277, 289
 - project contingencies, 105
 - rollbacks, 297
- Platform as a Service (PaaS), 462, 483, 489
- Pods, 328, 329, 330, 342, 391–392
- Pools, 328–329, 344–345, 347–348
- Portability between clouds, 472, 474
- Positive stress testing, 281
- Postmortems
 - after crises, 172–173, 175
 - leading, 153–156, 158
 - noticing incidents too late, 495–496
 - recognizing early signs of problems in, 496–499
- Power utilization of data centers, 510, 512, 513, 528, 529
- PPACA (Patient Protection and Affordable Care Act), 331–332
- Preparing cloud applications
 - applying Scale Cube in cloud, 485–487, 493, 494
 - fault isolation challenges, 487–489, 494
 - Intuit case study, 491–493
 - overview, 485, 493–494
 - variability in input/output, 489–491, 494
- Principles. *See* Architectural principles
- Problems
 - conflicts managing incidents and, 150
 - defined, 145–146, 158
 - detecting, 496–503, 507
 - developing monitors to detect, 502–503
 - identifying incidents from, 495–499
 - life cycles for, 150–151
 - locating indicators of, 501
 - management components of, 149–150
 - postmortems for, 153–156, 158
 - resolving, 147
- Processes. *See also* Barrier conditions
 - assigning ownership for team, 140, 142
 - automating, 221–222
 - choosing, 138–139, 141
 - CMMI framework for, 132, 134–135, 136–137
 - complexity of, 137–139, 141
 - continuous improvement of, 192, 195
 - creating crisis management, 163–164
 - culture clash with, 139–140, 141, 142
 - defined, 132
 - developing code without team, 295
 - identifying stress testing, 282, 286
 - marshalling and unmarshalling, 399
 - overview, 132, 140–142
 - value of, 131–132, 140–141
 - when to implement, 137, 141
- Proctor & Gamble, 99
- Product organization model, 122–124
- Production environments. *See also* Rollbacks
 - change identification in, 179–180, 181, 194
 - cloud computing uses for, 476–478
 - continuous delivery in, 182
 - markdown functionality for, 333
 - mimicking for performance testing, 275–276, 280, 289, 555, 556
 - simulating for stress testing, 283, 286

- Production grids, 454
 - Productivity
 - measuring, 12–13
 - organizational cost of scale, 14–17, 20
 - right behaviors and, 11
 - team size and poor, 47
 - Products. *See also* Features; Projects; Standards
 - automating processes, 221–222
 - determining headroom for, 197–208
 - drawbacks of stateful, 218–219
 - implementing search as own service, 389–391
 - JAD entry/exit criteria for features, 228–230
 - using IT model for customer, 122–124
 - Project triangle, 303–306, 313
 - Projects
 - fast or right tradeoffs in, 303–313
 - 5-95 Rule for, 104, 105, 117
 - management contingencies for, 104–105
 - managing, 102–105, 116
 - triangle of tradeoffs in, 303–306, 313
 - Pros-and-cons comparisons, 308–309, 313
 - PROS software systems, 368–370
 - Proulx, Tom, 99
 - Proxy caches, 402–403, 408
 - Proxy server, 402–403
 - Public vs. private clouds, 462–463
 - Pulling activities, 17, 19
 - Pushing activities, 17, 18
- Q**
- Quality. *See also* Tradeoffs
 - analyzing tradeoffs in, 307–311
 - defining project, 304–305
 - factoring in project triangle, 303–306, 313
 - measuring, 113
 - Quality assurance, 33
 - Quarterly incident reviews, 153, 157, 158
 - Query flexibility for NoSQL solutions, 442
 - Questions
 - asking, 25–26
 - build vs. buy, 255, 258, 321–322
 - evolving for system monitoring, 496–501, 507
 - regarding data center location, 513–514
 - QuickBooks, 99, 100
 - Quicken, 99, 100
 - Quicksort algorithm, 412
 - Quigo, 91, 92–94, 114–115, 210
- R**
- Rack units (U), 509–510
 - RAD (rapid application development), 296–297
 - RASCI acronym
 - applying to architectural principles, 214, 223
 - defining roles using, 36–39, 40, 156
 - Repeating performance tests, 279–280, 556, 561
 - Replication delays, 377, 380
 - Resonant Leadership* (Boyatzis and McKee), 76–77
 - Resources
 - example of resource contention, 412
 - further reading on scalability, 535
 - working with limited Agile, 242–243, 247
 - Response time measurements, 112
 - Reverse proxy caches, 403–405, 408, 409
 - Reviews. *See also* ARB
 - change, 191, 194, 195
 - code, 292, 296
 - quarterly incident, 153, 157, 158
 - 360-degree, 77, 96
 - Richter-Reichhelm, Jesper, 245–246
 - Right behaviors, 11
 - Right person, 10–11
 - Risk management
 - assessing risk, 268
 - continuous process improvement and, 192–195
 - human factors in, 270–271
 - importance in scalability, 259–261
 - measuring risk, 261–267
 - overview, 271–272
 - relation of performance and stress testing to, 288

- risk model, 260
- rules for acute, 268–270
- Risks
 - evaluating production tradeoffs, 307–311
 - FMEA (failure mode and effects analysis), 264–267, 268, 270–271
 - gut feel method, 261–263, 271, 308, 313
 - identifying for change, 185–186
 - managing acute, 268–270
 - measuring, 261–267
 - noting high-risk features, 235
 - plotting own/lease/rent options against, 517
 - risk assessment steps, 268
 - technology-agnostic design and, 320, 326
 - tradeoff rules for, 311, 313
 - traffic light method, 263–264, 268, 271
 - when changing schedules, 187–188
- Roadmap to success, 94–95
- Roles
 - clarity in team, 21–23, 40
 - defining, 23–24
 - engineering lead's, 167–168, 174
 - executives, 25–30
 - individual contributors, 30–34, 167–168, 174
 - missing skill sets and, 34–35
 - problem manager, 164–166, 174
 - RASCI matrix for defining, 35–39
 - responsibilities of, 35–39
 - team managers in crises, 166–167
- Rollbacks
 - checklist for, 298
 - costs of, 299–300
 - designing architecture for, 215, 222, 302
 - incorporating in change management, 190
 - planning for, 297
 - rollback insurance policy, 298, 299, 302
 - technical considerations for, 298–299
 - version numbers for, 299
 - window for, 297–298
- Rules
 - acute risk management, 268–270
 - Pareto 80/20 rule, 276
 - technology-agnostic design, 323–325
 - tradeoff, 311, 313
- Rules of Three
 - about, 215, 528–529
 - applying to data centers, 519–525
- S
- SaaS (Software as a Service)
 - capacity planning calculations for, 547–553
 - evolution of, 59, 60, 461, 462, 482
 - Intuit's development of, 99–100
 - load and performance calculations for, 555–561
 - projects using functional organizations, 53
 - using database splits with, 391–392
 - variability in cloud input/output, 489–491, 494
- Saban, Nick, 131
- SABRE (Semi-automated Business Research Environment) reservation system, 367–368
- Salesforce, 330–331, 391–392
- Scalability. *See also* AKF Scale Cube; Data storage; Headroom
 - Agile Organizations and, 60–61
 - art vs. science of, 4, 531–532
 - barrier conditions for, 292–293
 - build vs. buy decisions and, 249–250
 - business case for, 124–127, 128
 - caching and, 395, 409
 - cloud computing and, 459, 481–483
 - collaborative design for, 226–227
 - company needs for, 4
 - defining direction for, 90–94
 - eBay crisis in, 162–163
 - effect of crises on, 161
 - empowering, 82–83
 - fault isolation and, 334
 - grid computing in, 449–451, 458
 - headroom calculations and, 198–199, 208
 - implementing, 532–534
 - incidents related to, 144–145
 - issues in, 2–3

- Scalability (*continued*)
- management and leadership in, 17–19, 20, 82–83
 - managing changes, 177–178
 - NoSQL solutions for, 440–443, 445
 - organizational cost of scale, 14–17
 - organizational factors in, 11–17, 20, 41–44
 - people and, 10–11, 20, 531–532
 - processes in, 131–132
 - resources on, 535
 - risk management in, 259–261
 - scale on demand cloud computing, 464–465
 - scaling agnostically, 250
 - scaling out, not up, 219, 222
 - sports analogy for, 105–107
 - supporting with TAD, 321–323, 326
 - synchronous or asynchronous calls in, 414–415, 422–423
 - team failures in, 21–23
 - tradeoffs in, 306–307
 - using multiple axes of, 365–367
 - vicious/virtuous cycles in, 3, 532, 533
 - x-axis splits and, 359–361
 - y-axis splits and, 361–362
 - z-axis splits and, 363–364
- Scalability projects, 201
- Scheduling changes, 187–189, 194
- Scope
- effect on project triangle, 305
 - factoring in project triangle, 303–306, 313
 - scalability and, 306
- Sculley, John, 11
- Search as own service, 389–391
- Search engine marketing, 91
- Seattle Computer Products, 255–256
- Security in cloud computing, 471–472, 474
- Seed, feed, and weed activities, 107–111, 117
- Servers. *See also* Architectural principles; Performance testing; Stress testing
- applying Rule of Three to data center, 520–521, 528, 529
 - asynchronous system design for, 217–218, 222
 - capacity planning calculations for, 547–553
 - decreasing size of computers, 509–510
 - Etsy upgrades for, 239–240
 - headroom usage percentages for, 203–205
 - implementing reverse proxy, 404–405
 - monitoring requests for Web and application, 548–549
 - proxy cache implementation for, 402–403
 - reverse proxy caches for, 409
 - simulating environment for stress testing, 283, 286
- ServiceNow, 392
- Shard, 328, 329, 342
- Shared goals, 22–23, 103–104
- Shared infrastructure for grid computing, 450, 451, 452, 458
- Shareholders
- leader alignment to values of, 83, 96
 - leader's pursuit of value for, 79–80
- Silo organizations. *See* Functional organizational structure
- Skill sets, 34–35, 478
- Slivers, 328, 329
- Smart, Geoff, 108
- SMART goals, 89–90, 96, 97, 156, 211
- Smith, Adam, 244
- Software, caching, 406–407
- Software as a Service. *See* SaaS
- Software Engineering Institute, 132, 134
- Source code
- compilation steps for build grids, 455–456
 - cowboy coding, 295
 - introducing reviews of, 292, 296
 - KLOC, 113–114
 - measuring engineer's production of, 113–114
 - splitting into failure domains, 49, 50
- Speed. *See also* Tradeoffs
- analyzing tradeoffs in, 307–311
 - benefits of cloud computing, 470, 471
 - defining project, 305
 - factoring in project triangle, 303–306, 313

- Splits. *See also* AKF Scale Cube; Splitting applications; Splitting databases
Big Data, 438–440, 445
making team, 49–51
service- or resource-oriented, 436–437
using data center live sites, 521–525
within cloud environments, 485–487, 493, 494
- Splitting applications
effects of, 370–371
overview, 357, 371–373
x-axis for AKF Application Scale Cube, 357–358, 359, 361, 365–367, 371–372
y-axis for AKF Application Scale Cube, 357–358, 361–362
z-axis for AKF Application Scale Cube, 358, 363–364
- Splitting databases
computing headroom using *x*-axis splits, 550
using AKF Scale Cube for, 375–376
y- and *z*-axis data splits, 438–440, 445
- Spotify Agile team structure, 68–69, 244–245
- Squads, 68–69, 244–245
- Srivastava, Amitabh, 43
- Stability, defined, 177
- Standards
Agile methodologies and, 293
fostering within organizations, 42–43
maintaining across teams, 243–246, 248
ownership of product, 44
- Stansbury, Tayloe, 100, 102–103
- State machines, 419, 424
- Stateful applications
avoiding, 218–219
user sessions and, 420–422, 424
when to use, 351
- Stateless systems
advantages of, 218–219, 222, 351
using stateless session data, 420–421
- Statistical process control chart (SPCC), 500, 501
- Status communications, 171–172
- Steve Jobs* (Isaacson), 11
- Strategic competitive advantage, 430
- Strategic competitive differentiation, 432
- Strategy-focused approaches in build vs. buy decisions, 251, 258
- Stress testing
about, 281, 289
analyzing data from, 285, 286
creating load for, 284, 286
determining load for, 283, 286
establishing environment for, 283, 286
executing tests, 284–285, 286
key services in, 282, 286
objectives for, 281–282, 285
overview, 288–290
processes for monitoring, 284, 286
relating to scalability, 287–288, 290
steps in, 285–286, 289–290
- Structures for caching, 396, 409
- Success
causal roadmap to, 94–95, 97
managing path to, 115–116, 117, 118
- Success criteria for performance tests, 274, 280, 555, 556
- Sun Grid Engine, 448
- Sun Tzu, 9, 21, 41, 71, 99, 119, 131, 143, 159, 177, 197, 209, 225, 239, 259, 273, 291, 303, 317, 327, 343, 357, 375, 395, 411, 427, 447, 459, 485, 509, 531
- Swim lanes
along natural barriers, 340, 341
defined, 327–328, 329, 342
identifying recurring incidents for own, 339, 341
isolating money-makers within, 339, 340, 342
measuring service availability and, 542–543
no crossing boundaries of, 338, 342
transactions along, 338, 342
- Synchronization. *See also* Asynchronous design
methods of, 411–412, 423
mutex, 411, 423
using synchronous vs. asynchronous calls, 412–413, 414–415, 422–423
- Systems monitoring, 501–502

T

- TAA (technology-agnostic architecture), 318–319, 323–325
- TAD. *See* Technology-agnostic design
- Tags
 - controlling caching with meta, 405
 - tag-datum cache structure, 396, 399, 409
- Tasks, 102–105, 116
- Team size
 - checklist for, 50–51
 - constraints on, 45–46
 - employee experience and, 45
 - factors increasing, 46
 - growing or splitting teams, 48–51
 - managers and, 45–46
 - overview, 70
 - signs of incorrect, 47–48
- Teams. *See also* Team size
 - autonomy of Agile, 241–242, 247
 - boundaries of, 65
 - building, 105–107
 - business unit owners, general managers, and P&L owners, 27–28
 - capacity planning for, 33–34
 - CFOs of, 27–28
 - choosing processes for, 138–139, 141
 - collaboration between, 103–104
 - CTO/CIO of, 28–30
 - developing and using processes, 132–134
 - DevOps responsibilities, 31–32
 - egotist behavior when leading, 79–80, 96
 - engineers, 31–33
 - feeding members of, 108–109, 117
 - growing or splitting, 48–51
 - infrastructure, 32–33
 - interviewing new members, 108, 116
 - IT model for customer product, 122–124
 - leader’s empowering of, 82–83
 - maintaining standards across, 243–246, 248
 - management and path to success, 115–116, 117, 118
 - organization of Agile, 61
 - ownership of principles by, 213–214
 - quality assurance, 33
 - recognizing badly fitted processes, 139–140, 141, 142
 - relationships with management, 119–122, 128
 - roles in, 21–24, 35–39
 - scaling cross-functional designs, 226–227
 - seeding performance of, 107, 110, 117
 - selecting architectural principles, 212–213
 - sense of empowerment for, 64–65
 - sharing goals, 22–23
 - size and productivity of, 15
 - skill sets of, 34–35, 478
 - Spotify’s, 68–69, 244–245
 - structure of, 14
 - system architects, 30–31
 - team manager role, 166–167
 - tool for defining roles, 35–39
 - Two-Pizza Team rule, 16–17
 - upgrading, 107–111
 - weeding individuals from, 109–110, 117
 - Wooga cross-functional, 245–246
 - working with limited resources, 242–243, 247
- Technology. *See also* CTOs
 - calculating hardware uptime, 540–541
 - missing skill sets in, 34–35
 - vicious and virtuous cycles in, 3, 532
- Technology-agnostic architecture (TAA), 318–319, 323–325
- Technology-agnostic design (TAD)
 - about TAA and, 318–319
 - availability and, 323, 326
 - build vs. buy decisions and, 323, 325
 - costs of, 319–320, 326
 - risks and, 320, 326
 - rules for, 323–325
 - supporting scalability with, 321–323, 326
- Terminating employees, 109–110
- Testing. *See* Performance testing; Stress testing
- Theory of innovation model, 66
- Thin slicing, 262
- Thousands of lines of code (KLOC), 113–114

- Thrashing, 203–204
 - Threads
 - initiating new, 413, 423
 - swapping synchronously, 414
 - thread join synchronization, 412
 - 360-degree review process, 77, 96
 - Tiered storage solutions, 432–433, 444
 - Time
 - summarizing headroom, 202–203
 - time to market and fault isolation, 334–335
 - Tradeoffs
 - assessing feature, 230, 234, 235
 - factors in project triangle, 303–306, 313
 - fast or right checklist, 310–311
 - risks and rules for, 311, 313
 - weighing risks in fast or right, 307–311
 - Traffic light risk assessment, 263–264, 268, 271
 - Transactions
 - implementing eBay database splits for, 388–389
 - reducing time with y-axis database splits, 382
 - x-axis splits for growth in, 371
 - z-axis database splits for scaling growth of, 383–385
 - Transformational leadership, 84, 96
 - Transforming data, 433–434
 - Tribes, 69, 245
 - TurboTax, 99, 100, 492
 - Twitter Fail Whale, 197
 - “Two-Pizza” Rule, 16–17, 44
- U**
- UNICORE (UNiform Interface to Computing REsources), 448
 - Unit testing, 293
 - Unprofitable customers, 431
 - Upgrading teams, 107–111
 - U.S. Constitution
 - goals within, 89–90
 - mission outlined in, 88
 - vision of Preamble, 86
 - U.S. Declaration of Independence, 86
 - U.S. Pledge of Allegiance, 85–86
- User sessions**
- approaches to scaling in, 420–422, 424
 - avoidance in, 420–421, 422
 - centralization in, 421, 422
 - decentralization in, 421, 422
 - stateful applications and, 420
- Utilization plotted against own/lease/rent options, 518–519
- V**
- Validating changes, 189–190, 194
 - Venn diagrams, 212–213
 - Vicious/virtuous technology cycles, 3, 532
 - Virtualization for cloud computing, 466–467
 - Vision, 96
 - Vision statements
 - applying to scalability solutions, 92
 - developing, 84–87, 96–97
 - managing teams toward, 103–104
 - Von Moltke, Helmut, 104
- W**
- War rooms, 169–170, 175
 - Waterfall development models, 295–296
 - Web Operations* (Allspaw), 159
 - Web pages, 405, 408
 - Web servers, 548–549, 551
 - Web site availability, 539, 540–541
 - Who* (Smart), 108
 - Wilson, Mike, 389
 - Wooga, 244, 245–246
- X**
- X-axis**
- AKF Application Scale Cube, 357–358, 359, 361, 365–367, 371–372
 - AKF Database Scale Cube, 376–381, 386, 387, 393
 - AKF Scale Cube, 344–346, 351, 352, 354
 - cloud environment and Scale Cube, 485–486, 493, 494
 - when to use database splits, 393, 394
- XaaS (Everything as a Service), 461, 462, 483

Y

Y-axis

- AKF Application Scale Cube, 357–358, 359, 361–362, 365–367, 371, 372–373
 - AKF Database Scale Cube, 381–383, 386, 387, 388, 393–394
 - AKF Scale Cube, 346–348, 351, 352, 354
 - cloud environment and Scale Cube, 486–487, 494
 - scalability and Big Data splits, 438–440, 445
 - when to use database splits, 393, 394
- Yavonditte, Mike, 91

Z

Z-axis

- AKF Application Scale Cube, 357–358, 359, 361–362, 363–364, 365–367, 371–373
- AKF Database Scale Cube, 383–386, 387, 388, 394
- AKF Scale Cube, 349–350, 351, 352, 354
- cloud environment and Scale Cube, 486–487, 494
- scalability and Big Data splits, 438–440, 445
- when to use database splits, 393, 394