



# THE ART OF SCALABILITY

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

*"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**, Director of Engineering, Facebook

MARTIN L. ABBOTT

MICHAEL T. FISHER

FOREWORD BY BOB DAVIS, Managing Partner, Highland Capital Partners, and Founder/Former CEO, Lycos

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.

The publisher offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales, which may include electronic versions and/or custom covers and content particular to your business, training goals, marketing focus, and branding interests. For more information, please contact:

U.S. Corporate and Government Sales  
(800) 382-3419  
corpsales@pearsontechgroup.com

For sales outside the United States please contact:

International Sales  
international@pearson.com

Visit us on the Web: [informit.com/aw](http://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.  
p. cm.

Includes index.

ISBN-13: 978-0-13-703042-2 (pbk. : alk. paper)

ISBN-10: 0-13-703042-8 (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 2010

658.4'06—dc22

2009040124

Copyright © 2010 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. For information regarding permissions, write to:

Pearson Education, Inc.  
Rights and Contracts Department  
501 Boylston Street, Suite 900  
Boston, MA 02116  
Fax: (617) 671-3447

ISBN-13: 978-0-13-703042-2

ISBN-10: 0-13-703042-8

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

First printing, December 2009

**Editor-in-Chief**  
Mark Taub

**Acquisitions Editor**  
Trina MacDonald

**Development Editor**  
Songlin Qiu

**Managing Editor**  
John Fuller

**Project Editor**  
Anna Popick

**Copy Editor**  
Kelli Brooks

**Indexer**  
Richard Evans

**Proofreader**  
Debbie Liehs

**Technical Reviewers**  
Jason Bloomberg  
Robert Guild  
Robert Hines  
Jeremy Wright

**Cover Designer**  
Chuti Prasertsith

**Compositor**  
Rob Mauhar

# Foreword

In 1996, as Lycos prepared for its initial public offering, a key concern among potential investors of that day was whether our systems would scale as the Internet grew; or perhaps more frightening, would the Internet itself scale as more people came online? And the fears of data center Armageddon were not at all unfounded. We had for the first time in human history the makings of a mass communications vehicle that connected not thousands, not millions, but billions of people and systems from around the world, all needing to operate seamlessly with one another. At any point in time, that tiny PC in San Diego needs to publish its Web pages to a super computer in Taipei, while Web servers in Delhi are finding a path over the information highway to a customer in New York. Now picture this happening across billions of computers in millions of locations all in the same instant. And then the smallest problem anywhere in the mix of PCs, servers, routers, clouds, storage, platforms, operating systems, networks, and so much more can bring everything to its knees. Just the thought of such computing complexity is overwhelming.

This is exactly why you need to read *The Art of Scalability*.

Two of the brightest minds of the information age have come together to share their knowledge and experience in delivering peak performance with the precision and detail that their West Point education mandates. Marty Abbott and Mike Fisher have fought some of the most challenging enterprise architecture demons ever and have always won. Their successes have allowed some of the greatest business stories of our age to develop. From mighty eBay to smaller Quigo to countless others, this pair has built around-the-clock reliability, which contributed to the creation of hundreds of millions of dollars in shareholder value. A company can't operate in the digital age without flawless technical operations. In fact, the lack of a not just good, but great, scalable Web architecture can be the difference between success and failure in a company. The problem though, in a world that counts in nanoseconds, is that the path to that greatness is rarely clear. In this book, the authors blow out the fog on scaling and help us to see what works and how to get there.

In it, we learn much about the endless aspects of technical operations. And this is invaluable because without strong fundamentals it's tough to get much built. But when I evaluate a business for an investment, I'm not only thinking about its products; more importantly, I need to dig into the people and processes that are its foundation. And this is where this book really stands out. It's the first of its kind to examine the impact that sound management and leadership skills have in achieving scale. When systems fail and business operations come crashing down, many are

quick to look at hardware and software problems as the root, whereas a more honest appraisal will almost always point to the underlying decisions people make as the true culprit. The authors understand this and help us to learn from it. Their insights will help you design and develop organizations that stand tall in the face of challenges. Long-term success in most any field is the result of careful planning and great execution; this is certainly so with today's incredibly complex networks and databases. The book walks you through the steps necessary to think straight and succeed in the most challenging of circumstances.

Marty and Mike have danced in boardrooms and executed on the frontlines with many of the nation's top businesses. These two are the best of the best. With *The Art of Scalability*, they have created the ultimate step-by-step instruction book required to build a top-notch technical architecture that can withstand the test of time. It's written in a way that provides the granular detail needed by any technical team but that can also serve as a one-stop primer or desktop reference for the executive looking to stand out. This is a book that is sure to become must-reading in the winning organization.

*Bob Davis*

*Managing Partner, Highland Capital Partners, and Founder/Former CEO, Lycos*

# Introduction

This book is about the *art of scale, scalability, and scaling* of technology organizations, processes, and platforms. The information contained within 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 formulate the right scalability questions and focus on the right issue, whether that be people, process, or technology, in order to help prevent scalability disasters. For the technical executive, manager, or individual engineer, we address the organizational and process issues that negatively impact scale as well as provide technical models and advice to build more scalable platforms.

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 learned about scalability from having suffered through the challenges of scaling systems for a combined thirty plus years. We have been engineers, managers, executives, and advisors for startups as well as Fortune 500 companies. The list of companies that we have worked with includes familiar names such as General Electric, Motorola, Gateway, eBay, and PayPal. The list also includes hundreds of less-known startups that need to be able to scale as they grow. Having learned the scalability lessons through thousands of hours diagnosing problems and thousands more hours of designing preventions of those problems, we want to share this combined knowledge. This was the motivation behind starting our consulting practice, AKF Partners, in 2007 and continued to be the motivation for producing this book.

---

## 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 are often called an error chain and they often start with human rather than mechanical failure. In fact, Boeing identified that 55% of the aircraft incidents with Boeing aircraft between 1995 and 2005 had human factors related causes.<sup>1</sup>

---

1. Boeing (May 2006), "Statistical Summary of Commercial Jet Airplane Accidents Worldwide Operations."

Our experience with scalability-related issues follows a similar trend. The CTO or executive responsible for scale of a technology platform may see scalability as purely a technical endeavor. This is the first, and very human, failure in the error chain. As a result, the process to identify a need to split a database into multiple databases doesn't exist: failure number two. When the user count or transaction volume exceeds a certain threshold, the entire product fails: failure number three. The team assembles to solve the problem and because it has never invested processes to troubleshoot problems such as these the team misdiagnoses the failure as "the database just 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, through the application, to the database, and even pointing fingers at each other. Customers walk away, morale flat lines, and shareholders are left holding the bag.

The 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 business and technology executives and advisors, scalability issues start with organizations and people and 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 0.1. 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 0.1.

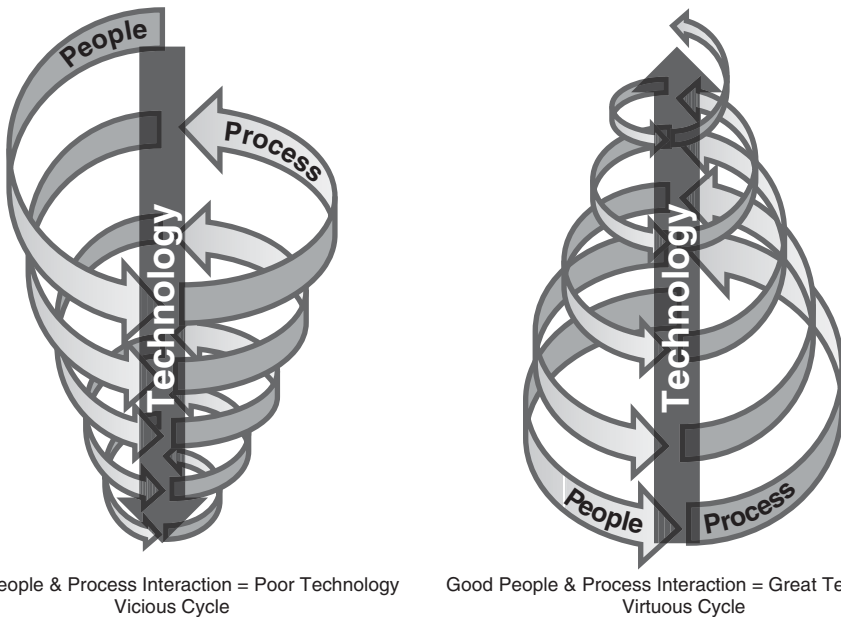
---

## Art Versus Science

The use of the word *art* is a very deliberate choice on our part. Besides fitting nicely into the title and allowing us to associate some of Sun Tzu's teachings into our own book, Merriam-Webster's dictionary gives one definition of art as a "branch of learning."<sup>2</sup> Additional definitions offered by Merriam-Webster are "skill acquired by experience, study, or observation" and "an occupation requiring knowledge or skill." All of these are true of the nature of scaling platforms, processes, and organizations. But perhaps more important in our choice of *art* here are the images the word conjures up of being more fluid versus the view of science, which is more structured and

---

2. Merriam-Webster Online Dictionary. <http://www.merriam-webster.com/dictionary/art>.



**Figure 0.1** *Vicious and Virtuous Technology Cycles*

static. It is this image that we are relying upon heavily within our title 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. Rather, the interactions between platforms, organizations, and processes have profound impacts on the adaptation of any specific and highly structured approach. The 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. Consistent with this use of *art*, our book focuses on providing skills and lessons regarding approaches rather than improperly teaching that a one-size-fits-all approach will solve any need.

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 cookie cutter approach for 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 systems

through much of this book, we do so only because the greatest unprecedented growth has been experienced by Internet companies like Google, Yahoo, eBay, Amazon, Facebook, and the like. But many other companies have 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 collocation facility (data center) providers. We guarantee that scalability was on the mind of every bank 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 and they work not only in front-end customer facing transaction processing systems but 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 your business to operate without concerns regarding customer or end-user demand.

---

## Book Organization and Structure

We've divided the book into four 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 often aren't given the time or resources to develop business, management, and leadership acumen. Although they might perform well in the architectural and technical aspects of scale, their expertise in organizational scale needs are often shallow or nonexistent. Our intent is to arm 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 our consulting and advisory practice of AKF Partners. These models 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 what to consider when planning data centers and how to evolve your monitoring strategies to be closer to your customers.

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 a float. 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, other times as observers. We have seen how putting these lessons into practice has yielded success and the unwillingness or inability to do so has led to failure. This book will teach these lessons and hopefully put you and your team on the road to success. We believe the lessons herein are valuable for everyone from engineering staffs to product staffs including every level from the individual contributor to the CEO.

# Chapter 1

---

---

# The Impact of People and Leadership on Scalability

Fighting with a large army under your command is nowise different from fighting with a small one; it is merely a question of instituting signs and signals.

—Sun Tzu

People, organizational structure, management, and leadership all have an impact on the scalability of your organization, your processes, and (as a result) the scalability of your product, platform, or systems. They are at the heart of everything you do and the core of everything you need to scale a company and a platform. Paradoxically, they are the things we overlook most often when attempting to scale large systems: Our people are overlooked and underappreciated; organization structure is a once-a-year, check-the-box exercise written in haste in PowerPoint and managed by HR; and our managers and leaders are often untrained or undertrained in the performance of their duties. In this chapter, we will explain why the people of your organization, the structure of the organization, the management, and the leadership in your organization all have an enormous impact on your ability to scale your product, platform, or services.

---

## Introducing AllScale

Throughout *The Art of Scalability*, we will refer to a fictional company, AllScale. AllScale started out as a custom software development company, contracting individual developers out by the hour for projects. Over time, the company started to bid on special custom development projects for both back office IT systems and Web enabled Software as a Service (SaaS) platforms. As the company matured, it started developing tools for its own internal usage and then started selling these tools as a service to other companies using the SaaS model.

The tool with which AllScale has had the most traction is the human resources management (HRM) system. The tool is an employee life cycle management system, covering everything from recruiting to termination. The recruiting process is automated, with resumes held online and workflows depicting the status of each recruit and notes on the interview process. After an employee is hired, all corporate training material is performed online through the system. Employee reviews are performed within the system and tracked over time. Associated merit increases, notes from one-on-one sessions, previous jobs, and performance information are all contained within the system. When an employee leaves, is terminated, or retires, the notes from the exit interview are retained within the system as well.

AllScale is a private company with a majority ownership (51%) obtained by a single venture capital (VC) company after a B-series round. The VC firm invested in both rounds, having decided to make its initial investment after the company started building SaaS product offerings and seeing how AllScale's HRM software started to rapidly penetrate the market with viral adoption.

AllScale is an aggregation of our experience with our clients and our experience running technology organizations within Fortune 500 and startup companies. We decided to focus on one imaginary company for the sake of continuity across people, process, and technology issues. The evolution of AllScale from job shop contractor to the developer of multiple SaaS offerings also allows us to take a look at several unique challenges and how the management team might overcome them.

---

## Why People

In our introduction, we made the statement that people are important when attempting to scale nearly anything; they are especially important when trying to scale technical platforms responsible for processing transactions under high user demand and hyper growth.

Here, we are going to go out on a limb and assert that people are the most important aspect of scale. First and foremost, without people, you couldn't possibly have developed a system that needs to scale at all (at least until such point as the HAL 9000 from *2001: A Space Odyssey* becomes a reality). Without people, who designed and implemented your system? Who runs it? Following from that, people are the source of the successes and failures that lead to whatever level of scale you have achieved and will achieve. People architect the systems, write or choose the software, and they deploy the software payloads and configure the servers, databases, firewalls, routers, and other devices. People make the tradeoffs on what pieces of the technology stack are easily horizontally scalable and which pieces are not. People design (or fail to design) the processes to identify scale concerns early on, root cause scale related availability events, drive scale related problems to closure, and report on scale

needs and initiatives and their business returns. Initiatives aren't started without people and mistakes aren't made without people. People, people, people . . .

All of the greatest successes in building scalable systems have at their heart a great set of people making many great decisions and every once in awhile a few poor choices. Making the decision not to look at people as the core and most critical component of scaling anything is a very large mistake and a step in a direction that will at the very least make it very difficult for you to accomplish your objectives.

As people are at the heart of all highly scalable organizations, processes, and systems, doesn't it make sense to attract and retain the best people you can possibly get? As we will discuss in Chapter 5, Management 101, it's not just about finding the people with the right and best skills for the amount you are willing to pay. It's about ensuring that you have the right person in the right job at the right time and with the right behaviors.

The VC firm backing AllScale has a saying amongst its partners that the "fish rots from the head." Although the firm's representative on AllScale's board of directors and the remainder of the board feel that the current CEO and founder did a great job of growing the company and identifying the HRM market opportunity, they also know that the competencies necessary to run a successful SaaS company aren't always the same as those necessary to run and grow a successful consulting company. After several quarters of impressive growth in the HRM market, the board becomes concerned over stalling growth, missed numbers, and a lack of consistent focus on the HRM product. The board brings in a seasoned SaaS veteran as the new CEO, Christine E. Oberman, and moves the previous founder and CEO to the position of chief strategy officer. Christine promises to bring in and retain the best people, structure the company for success along its current and future product offerings, and focus on management and leadership excellence to supercharge and maximize shareholder wealth.

The *right person* speaks to whether the person has the right knowledge, skills, and abilities. Putting this person in the right job at the right time is about ensuring that he or she can be successful in that position and create the most shareholder value possible while tending to his or her career and offering the things that we need to feel good about and comfortable in our jobs. The *right behaviors* speaks to ensuring that the person works and plays well with others while adhering to the culture and values of the company. Bad behaviors are as good a reason for removing a person from the team as not having the requisite skills, because bad behavior in any team member creates a vicious cycle of plummeting morale and productivity.

---

## Why Organizations

It should follow that if people are important to the scalability of a system, their organizational structure should also be important. If this isn't intuitively obvious, we

offer a few things to consider regarding how organizational structure and responsibilities can positively or negatively impact your ability to scale a system.

An important concept to remember when considering organizational design as it relates to scale or any situation is that there rarely is a single right or wrong organizational structure. Once again, this is an art and not really a science. Each organizational structure carries with it pros and cons or benefits and drawbacks relative to the goals you wish to achieve. It's important when considering options on how to structure your organization to tease out the implicit and explicit benefits and drawbacks of the organizational design relative to your specific needs.

Some questions you should ask yourself when developing your organizational design are

- How easily can I add or remove people to/from this organization? Do I need to add them in groups, or can I add individual people?
- Does the organizational structure help or hinder the development of metrics that will help measure work done by the organization?
- How easily is this organization understood by the internal and external stakeholders of the organization (i.e., my customers, clients, vendors, etc.)?
- How does this organizational structure minimize the amount of work I lose on a per person basis as I add people to the organization?
- What conflicts will arise within the organizational structure as a result of the structure and how will those conflicts hinder the accomplishment of my organization's mission?
- Does work flow easily through the organization or is it easily contained within a portion of the organization?

These aren't the only questions one should ask when considering organizational structure, but each has a very real impact to the scalability of the organization. The question of how easily people are added is an obvious one as it is very difficult to significantly increase the amount of work done by an organization if your organizational structure does not allow the addition of people to perform additional or different types of work. Additionally, you want the flexibility of adding people incrementally rather than in large groups and the flexibility of easily removing people as market situations demand, such as a sudden increase in demands on the company or a market recession requiring constriction of expenses.

The question regarding metrics is important because while you often need to be able to scale an organization in size, you also want to ensure that you are measuring the output of both the organization and the individual people within the organization. An important point to remember here is that as you add people, although the total output of the team increases, the average output per person tends to go down slightly. This is the expected result of the overhead associated with communication

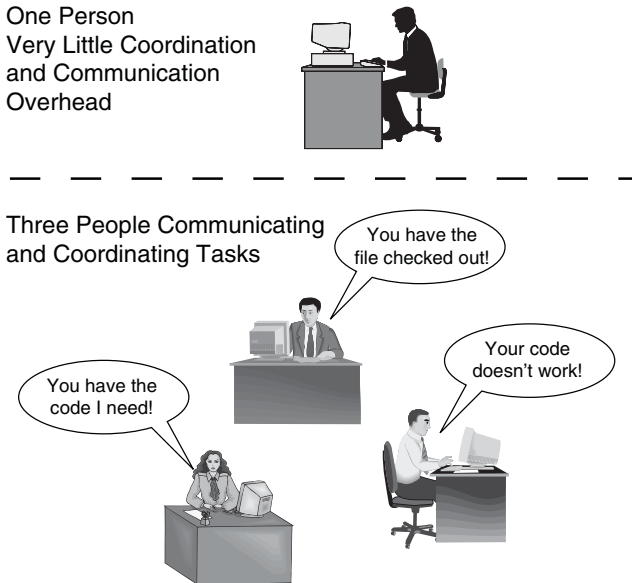
between people to accomplish their tasks. Each person can only work so many hours in a day and certainly no more than 24. If an organization consisting of a single person were to work the maximum possible hours in a day, constrained either by law or exhaustion, doing his or her primary task and absolutely nothing else, it stands to reason that the same person when required to interface with other people will have less time to accomplish his or her primary task and as a result produce less in the same amount of time. Therefore, the more people with whom an individual needs to interface to complete any given task, the more time it will take for that person to complete that task as increasing amounts of time are spent interfacing and decreasing amounts of time are spent performing the task.

The way to envision this mathematically is that if a single person can produce 1.0 unit of work in a given timeframe, a two-person organization might produce 1.99 units of the same work in the same timeframe. Each person's output was slightly reduced and while the team produced more overall, each person produced slightly less on an individual basis. The resulting relative loss of .01 units of work in the aforementioned timeframe represents the inefficiencies caused by coordination and communication. We will cover this concept in more detail in Chapter 3, *Designing Organizations*, where we discuss team size and how it impacts productivity, morale, and customer relations.

If the structure of your organization is such that it disallows or makes difficult the establishment of measurements on individual performance, you will not be able to measure output. If you cannot measure the output of individuals and organizations, you can't react to sudden and rapid deteriorations in that output resulting from an increase in size of the organization or a change in organizational structure.

"How easily is this organization understood by the internal and external stakeholders of the organization" addresses the need for intuitive organizational constructs. Written another way, this question becomes "Are you aligned with your stakeholders or do you waste time getting requests from stakeholders to the right teams?" If you want an organization to scale well and easily, you don't want the external teams with which you interface (your customers, vendors, partners, etc.) to be scratching their heads trying to figuring out with whom they need to speak. Worse yet, you don't want to be spending a great deal of time trying to figure out how to parcel work out to the right groups based on some stakeholder request or need. This might mean that you need to develop teams within your organization to handle external communication or it might mean that teams are developed around stakeholder interests and needs so that each external interface only works with a single team.

We discussed the question of "How does this organization structure minimize the amount of work I lose on a per person basis as I add people to the organization?" within our explanation of our question on metrics. You might have been in organizations where you receive hundreds of internal emails a day and potentially dozens of meeting invites/requests a week. If you've been in such a situation, you've no doubt spent



**Figure 1.1** *Coordination Steals Individual Productivity*

time just to eliminate the emails and requests that aren't relevant to your job responsibilities. This is a perfect example of how as you add people, the output of each individual within an organization goes down (refer back to our example of one person producing 1.0 unit of work and 2 producing 1.99 units of work). In the preceding example, as you add people, the email volume grows and time dedicated to reading and discarding irrelevant emails goes up. Figure 1.1 is a depiction of an engineering team attempting to coordinate and communicate and Table 1.1 shows the increase in overall output, but the decrease in individual output between an organization of three individuals and an organization consisting of one individual. In Table 1.1, we show an individual loss of productivity due to communication and coordination of .005, which represents 2.4 minutes a day of coordination activity in an 8-hour day. This isn't a lot of time, and most of us intuitively would expect that three people working on the same project will spend at least 2.4 minutes a day coordinating their activities even with a manager! One person on the other hand need not perform this coordination. So, as individual productivity drops, the team output still increases.

**Table 1.1** *Individual Loss of Productivity as Team Size Increases*

Organization Size	Communication and Coordination Cost	Individual Productivity	Organization Productivity
1	0	1	1
3	0.005	0.995	2.985

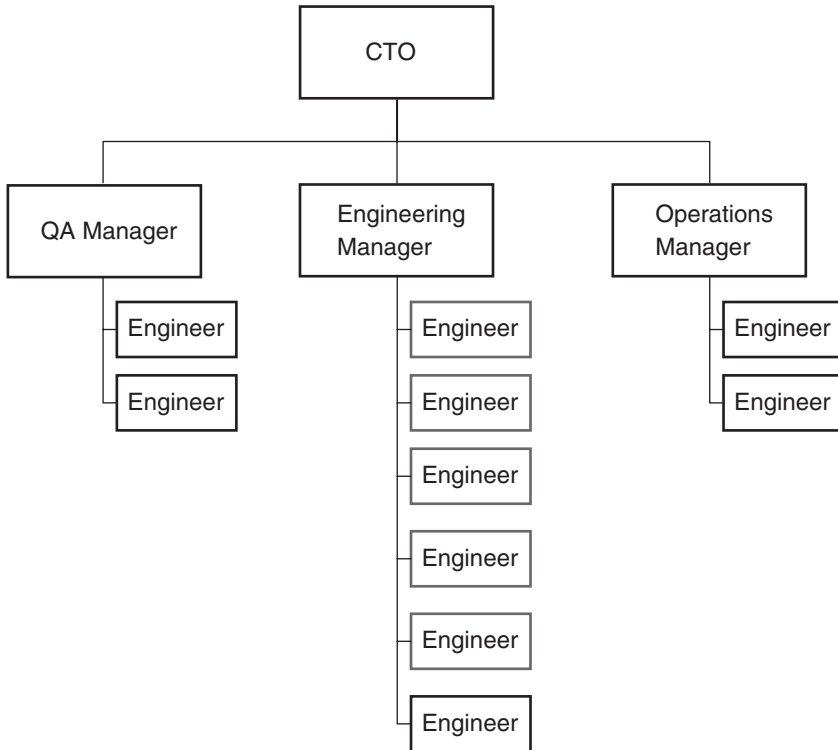
You can offset but not completely eliminate this deterioration in a number of ways. One possibility is to add management to limit interpersonal coordination. Another possibility is to limit the interactions between individuals by creating smaller self-sufficient teams. Both of these approaches have benefits and drawbacks that we will discuss in Chapter 3. Many other approaches are possible and anything that increases individual throughput without damaging innovation should be considered.

Another important point in organizational design and structure is that anywhere you create organizational or team boundaries, you create organizational and team conflict. The question “What conflicts will arise within the organizational structure as a result of the structure and how will those conflicts hinder the accomplishment of my organization’s mission?” attempts to address this problem, but there is really no way around boundaries causing friction. Your goal then should be to minimize the conflict created by organizational boundaries. The greatest conflict tends to be created when you have organizations with divergent missions, measurements, and goals, and an easy fix to this drawback is to ensure that every organization shares some set of core goals that drive their behaviors. We’ll discuss this in more detail in Chapter 3 where we will cover the two basic types of organizational structures and what purposes they serve.

“Does work flow easily through the organization or is it easily contained within a portion of the organization?” is meant to focus on the suitability of your organizational design to the type of work you do. Does work flow through your organization as efficiently as a well-defined assembly line? Does the type of work you do lend itself easily to a pipeline, where one team can start its work at a predefined place marked by where another team completes its work without a lot of communication overhead? Or is the work largely custom and highly intellectual, requiring a single team to work on it from start to finish without interruption? Are the components of what you build or produce capable of operating through a well-defined interface such that two teams can work on subcomponents at the same time?

Let’s take a look at our company, AllScale. AllScale recognizes that it has a need to scale the number of people within the engineering team that is supporting the HRM software in order to produce more products. Over the course of the last year, AllScale has added several engineers and now has a total of three managers and ten engineers. Each of the three managers reports to the chief technology officer (CTO) of AllScale. These engineers are broken down into the following teams:

- Two engineers responsible for the provisioning of systems, networking devices, databases, etc. for AllScale’s HRM product. This is the Operations team.
- Six engineers responsible for developing the applications that make revenue for AllScale’s HRM product. This is the Engineering team.
- Two engineers responsible for testing AllScale’s HRM product for defects and other quality related issues. This is the QA team.



**Figure 1.2** *AllScale Org Chart*

At a high level, we can intuit a few things from the structure of this organization. The designer of the organization believes that the separation into teams by skill set or functional job responsibility will not have an adverse impact on his or her ability to develop and launch new product functionality. The designer evidently sees great value in dedicating a group of people to testing the product to ensure it conforms to the company's quality standards. Benefits we would expect from such an organization are the ability to recruit top talent with focused skill sets such as software engineering in one or more programming languages, hardware/infrastructure experience, and quality/testing experience. At a high level, it appears that we should be able to relatively easily add engineers, operations/infrastructure engineers, and quality assurance engineers—at least until a manager is saturated with direct reports. This organization should be easily understood by all of the stakeholders as it is structured by relatively easily understood skills. Finally, work would seem to be able to flow easily between the organizations as we should be able to define measurable criteria that will qualify any given work product as being “ready” for the next phase of work. For instance, code might be ready for QA after it has passed a peer review and all unit testing is completed, and it might be ready for launching to the site and the Opera-

tions team after all priority one bugs are fixed and at least 90% of all other defects found in the first pass are resolved.

There are some potential drawbacks of such an organizational structure, however. For instance, how are you going to measure the throughput of the teams? Who is responsible for causing a slowdown of new initiative (feature or product) development? Will you measure your operations/infrastructure team by how many new features are launched to the site; if not, what keeps them from slowing down feature development in an attempt to increase a metric they will likely covet such as availability? When do you determine that something is “completed” for the purposes of measuring your engineering throughput? Is it when the feature launches live to site and if so, have you calculated the bug induced rework time in developing the feature?

Will the structure minimize the work loss on a per person basis as you grow the team? To know this, we probably need to dig into exactly how the software engineers are structured but we can probably also guess that coordination across teams is going to be a source of some work. Who will perform this coordination? Are the managers responsible for shepherding something from engineering to QA (Quality Assurance) and finally into the production team (Operations)? Who is responsible for setting the guidelines and criteria for when something moves from one place to another? Should you create a project management team responsible for helping to do this or should you instead reorganize your teams into self-contained teams that have all the skill sets necessary to complete any given task?

There are likely to be a great many conflicts in this proposed structure, many of them across the organizational boundaries we’ve defined. Operations will likely have concerns over the quality of new code or systems deployed, QA is likely to have concerns over the level of quality initially presented to them by Engineering, and Engineering will complain that Operations does not meet their needs quickly enough with respect to the creation of new systems, installation of new databases, and provisioning of new network devices. Who will be responsible for helping to resolve these conflicts, as each conflict takes time away from doing “real work.”

Other, larger questions we might have of such an organizational structure might be “Who is responsible for ensuring that the product or platform has an appropriate level of scale for our needs?” or “Who is responsible for identifying and resolving issues of scale?” When considering the answer to this question, please note that a scale issue might be the result of a network capacity constraint, a database capacity constraint, or a software capacity constraint. Moreover, that constraint isn’t going to be easily bucketed into one of these areas every time it comes up.

---

## Why Management and Leadership

In our experience, relatively few managers and leaders have ever had a course on management or leadership. Few universities offer such classes, unless you happen to

have been a management major or have attended an MBA program with a management curriculum. Given the lack of management and leadership courses in our universities, most people learn how to manage and how to lead informally: you watch what others do in peer positions and positions of greater responsibility and you decide what works and what doesn't. Over time, we start to develop our own "tool-boxes" and add tools from our professional readings or discard tools as they age and become less relevant to our younger generations of employees. This general "life as a lab" approach is how we've developed managers for years and, although it has its benefits, it is unfortunate that the two areas don't get better treatment in structured curriculums within universities and within larger corporations.

Management and leadership either multiply or detract from your ability to scale organizations in growth environments. They are often spoken of within the same context, but they are really two very different disciplines with very different impact on scalability. Many times, the same person will perform both the functions of a leader and a manager. In most organizations, one will progress from a position of an individual contributor into a primarily management focused role; and over time with future promotions, that person will take on increasing leadership responsibilities.

In general and at a very high level, you can think of management activities as "pushing" activities and leadership as "pulling" activities. Leadership sets a destination and "waypoints" toward that destination; management gets you to that destination. Leadership would be stating "We will never have a scalability related downtime in our systems" and management would be ensuring that it never happens. You absolutely need both and if you are going to scale your organization, your processes, and your systems well and cost effectively, you need to do both well.

Far too often, we get caught up in the notion of a "management style." We might believe that a person's "management style" makes them more of a leader or more of a manager. This notion of style is our perception of an individual's bias toward the tasks that define either leadership or management. We might believe that a person is more operationally focused and is therefore more of a "manager" or more visionary and therefore more of a "leader." Although we all have a set of personality traits and skills that likely make us more comfortable or more capable with one set of activities over the other, there is no reason we can't get better at both disciplines. Recognizing that they are two distinct disciplines is a step toward isolating and developing both our management and leadership capabilities to the benefit of our shareholders.

As we have indicated, management is about "pushing." Management is about ensuring that people are assigned to the appropriate tasks and that those tasks are completed within the specified time interval and at an appropriate cost. Management is about setting individual contributor goals along the path to the greater leadership goals and helping a team to accomplish both the individual contributor and team goals. It is also about ensuring that people get performance-oriented feedback in a

timely manner and that the feedback includes both praise for great performance and information regarding what they can improve. Management is about measuring and improving everything that ultimately creates shareholder value, examples of which are reducing the cost to perform an activity or increasing the throughput of an activity at the same cost. Management is communicating status early and often and clearly identifying what is on track and where help is needed. Management activities also include removing obstacles or helping the team over or around obstacles where they occur on the path to an objective. Management is important to scale as it is how you get the most out of an organization, thereby reducing cost per unit of work performed. The definition of how something is to be performed is a management responsibility and how something is performed absolutely impacts the scale of organizations, processes, and systems.

Management as it relates to people is about the practice of ensuring that we have the right person in the right job at the right time with the right behaviors. From an organizational perspective, it is about ensuring that the team operates well together and has the proper mix of skills and experiences to be successful. Management as applied to an organization's work is about ensuring that projects are on budget, on time, and meeting the expected results upon which their selection was predicated. Management means measurement and a failure to measure is a failure to manage. Failing to manage in turn is a guarantee to miss your organizational, process, and systems scalability objectives as without management, no one is ensuring that you are doing the things you need to do in the timeframe required.

Leadership has to do with all the pulling activities necessary to be successful in any endeavor. If management is the act of pushing an organization up a hill, leadership is the selection of that hill and then being first up it to encourage your organization to follow. Leadership is about inspiring people and organizations to do better and hopefully great things. Leadership is creating a vision that drives people to do the right thing for the company. Leadership is creating a mission that helps codify the aforementioned vision and creating a causal mental roadmap that helps employees understand how what they do creates value for the shareholder. Finally, leadership is about the definition of the goals on the way to an objective. Leadership is important to scale as it not only sets the direction (mission) and destination (vision) but it inspires people and organizations to achieve that destination.

Any initiative lacking leadership (including initiatives meant to increase the scalability of your company), while not doomed to certain failure, will likely only achieve success through pure dumb luck and chance. Great leaders create a culture focused on ensuring success through highly scalable organizations, processes, and products. This culture is supported by incentives structured around ensuring that the company scales cost effectively without user perceived quality of service or availability issues.

## Conclusion

We've asserted that people, organizations, management, and leadership are all important to scalability. People are the most important element of scalability, as without people there are no processes and there is no technology. The effective organization of your people will either get you to where you need to be faster or hinder your efforts in producing scalable systems. Management and leadership are the push and pull, respectively, in the whole operation. Leadership serves to inspire people to greater accomplishments, and management exists to motivate them to the objective.

## Key Points

- People are the most important piece of the scale puzzle.
- The right person in the right job at the right time and with the right behaviors is essential to scale organizations, processes, and systems.
- Organizational structures are rarely “right or wrong.” Any structure is likely to have pros and cons relative to your needs.
- When designing your organization, consider
  - The ease with which you can add people to the organization
  - The ease with which you can measure organizational success and individual contributions over time
  - How easy the organization is to understand for an outsider
  - How the organizational structure impacts individual productivity
  - What “friction” will exist between teams within the organization
  - How easily work flows through the organization
- Adding people to organizations may increase the organizational throughput, but the average production per individual tends to go down.
- Management is about achieving goals. A lack of management is nearly certain to doom your scalability initiatives.
- Leadership is about goal definition, vision creation, and mission articulation. An absence of leadership as it relates to scale is detrimental to your objectives.

# Index

13th Amendment, U.S. Constitution, 80  
360-degree reviews of leaders, 69

## A

Abbott, Marty, 505–506  
Abolition of slavery, goal example, 80  
“Above the Clouds: A Berkeley View . . .”, 447  
Accountable persons, RASCI, 38  
Achievable architectural principles, 198  
ACID (Atomicity, Consistency, Isolation, Durability) database properties, 383  
Action Phase, postmortem review, 145  
Action plans, creating, 504  
Acute risk, 252–253  
Adding  
    data to caches, 382  
    people to an organization. *See* Staffing.  
Adopting scalability, check lists and guidelines, 501–504  
After action review meeting. *See* Postmortem.  
Agile development, effects of barrier conditions, 275–277  
Agile Manifesto, effects of barrier conditions, 275–277  
Air traffic control, change case study, 170  
Aircraft incidents, causes of, 1  
AKF Scale Cube  
    axes, point of intersection, 327  
    cloning services and data, no bias, 328–330. *See also* X-axis.  
    concepts *vs.* rules, 325–326  
    initial point, 327  
    interpreting, 329  
    introduction, 326–327  
    scaling with increased data, 330  
    state within applications, 335  
    summary of, 334–336  
    uses for, 336–337  
    x-axis, 328–330, 334–335  
    y-axis, 331–332, 335  
    z-axis, 333–334, 335

AKF Scale Cube, splitting work responsibility by action. *See* Y-axis.  
    data. *See* Y-axis.  
    requestor or customer, 333–334. *See also* Z-axis.  
    resources, 331–332. *See also* Y-axis.  
    responsibility. *See* Y-axis.  
    services, 331–332. *See also* Y-axis.  
AKF Scale Cube for applications  
    application state, 341  
    cloning services, no bias, 341–342  
    code complexity, 346  
    configuration management, 342  
    handling codebase or database increases, 343–344  
    handling increased data, 342  
    introduction, 339–340  
    persistency, 341  
    scaling for transactions, 341–342  
    separating work responsibility, 343–344  
    splitting by customer or requestor, 344–347  
    time to market, 346  
AKF Scale Cube for applications, cost  
    maximum cost effectiveness, 349  
    x-axis splits, 341–342  
    y-axis splits, 344  
    z-axis splits, 345–347, 347–348  
AKF Scale Cube for applications, uses for  
    application complexity. *See* Y-axis splits.  
    back office IT systems, 352–353  
    customer base, growth. *See* Z-axis splits.  
    ecommerce, 350–351  
    ERP (enterprise resource planning), 351–352  
    human resources ERP systems, 351–352  
    observations, 353–354  
    transaction growth. *See* X-axis splits; Y-axis splits.  
    work growth, by system or platform. *See* X-axis splits; Y-axis splits.

- AKF Scale Cube for applications, x-axis splits
  - cost, 341–342
  - description, 341–342
  - fault isolation, 343
  - observing results, 353
  - uses for, 354
- AKF Scale Cube for applications, y-axis splits
  - cost, 344
  - description, 343–344
  - fault isolation, 345
  - observing results, 353
  - uses for, 354
- AKF Scale Cube for applications, z-axis splits
  - cost, 345–347
  - description, 344–347
  - fault isolation, 346
  - observing results, 353
  - uses for, 354
- AKF Scale Cube for databases
  - cloning data, no bias, 358–362. *See also* AKF Scale Cube for databases, x-axis splits.
  - distributed object caches, 360
  - introduction, 357–358
  - replication delays, 359–360
  - separating data into schemas, 362–365. *See also* Y-axis splits, databases.
  - splitting by customer geography, 365–367. *See also* Z-axis splits, databases.
  - splitting by requestor or customer, 365–367. *See also* Z-axis splits, databases.
  - summary of, 367–370
- AKF Scale Cube for databases, uses for
  - back office IT systems, 372–373
  - ecommerce, 370–372
  - ERP (enterprise resource planning), 372
  - human resources ERP, 372
  - observations, 373
  - timeline considerations, 373–374
- AKF Scale Cube for databases, x-axis splits
  - capacity planning, 361
  - configuration management, 361
  - cost, 360, 361–362
  - data consistency, 360–362
  - data currency, 362
  - description, 358–359
  - increasing data size or amount, 361
  - pros and cons, 360–362
  - reliance on third parties, 360–362
  - replication delays, 359–360, 361
  - summary of, 367–370
  - time to market, 360–361
  - vs.* y-axis splits, 363
- AKF Scale Cube for databases, y-axis splits
  - cost, 363, 364
  - description, 362–365
  - handling size and complexity, 364
  - noun perspective, 363
  - pros and cons, 363–364
  - purpose of, 362–363
  - summary of, 367–370
  - transaction processing time, 364
  - vs.* x-axis splits, 363
- AKF Scale Cube for databases, z-axis splits
  - cost, 366, 367
  - description, 365–367
  - pros and cons, 366
  - summary of, 367–370
- Allchin, Jim, 45
- Amazon outage example, 116
- Amazon.com Inc., cloud computing, 430
- Ambiguity, roles and responsibilities, 22–23
- Amdahl, Gene, 429
- Amdahl's Law, 429
- AMI (asynchronous method invocation), 395–396
- Analyzing data
  - performance testing, 261–262
  - stress testing, 268
- Annual budget, determining headroom, 184
- Apologizing to customers, 162–163
- Applications
  - caches. *See* Caches, application.
  - complexity, scaling for. *See* Y-axis splits.
  - monitoring, 477–478
  - splitting for scale. *See* AKF Scale Cube for applications.
  - state, AKF Scale Cube for applications, 341
- Approval
  - by ARB, 226, 229. *See also* ARB (Architecture Review Board); JAD (Joint Architecture Design).
  - change management, 174

- ARB (Architecture Review Board). *See also* JAD (Joint Architecture Design).
- approval, 226, 229
  - approving JAD proposals, 219
  - architectural principles, 222
  - as barrier conditions, 274
  - board constituency, 223–225
  - candidates for the board, 224
  - checklist for success, 229–230
  - conditional approval, 226, 229
  - conducting meetings, 225–227
  - entry criteria, 228–229
  - exit criteria, 228–229
  - feature selection, 228
  - image storage feature, example, 227–228
  - meeting structure, 225–227
  - pool of board candidates, 224–225
  - possible outcomes, 226, 229
  - purpose of, 222–223
  - rejection, 226, 229
  - rejection of components, 226, 229
  - rotating board positions, 225
  - tradeoffs, documenting, 228
- Architects, roles and responsibilities, 33
- Architectural principles
- achievable, 198
  - ARB (Architecture Review Board), 222
  - designing for any technology. *See* TAD (technology agnostic design).
  - desirable characteristics, 198
  - development process, 199–200
  - fault isolation. *See* Fault isolation, design principles.
  - following SMART guidelines, 198
  - goal tree, example, 196
  - goals of, 196–198
  - identifying and ranking, 199
  - JAD (Joint Architecture Design), 218
  - measurable, 198
  - ownership, 199–200
  - realistic, 198
  - specific, 198
  - testable, 198
  - Venn diagram of, 196
- Architectural principles, AKF recommendations
- asynchronous design, 202, 205–206
  - backward compatibility, 201
  - build *vs.* buy, 203
  - buy when non core, 203
  - checklist of, 208
  - designing to be disabled, 201
  - at least two axes of scale, 203, 207–208
  - monitoring, 202, 204
  - multiple live sites, 202, 205
  - N+1 design, 200–201
  - rollback, 201
  - rule of three, 200–201
  - scale out not up, 203, 207
  - stateless systems, 202, 206–207
  - use commodity hardware, 203
  - using mature technologies, 202
- Architecture
- vs.* implementation, 300
  - roles and responsibilities, 29–30
- Architecture Review Board (ARB). *See* ARB (Architecture Review Board).
- Art of scalability, 2–3
- Artificial Intelligence, 427–428
- Assessing your situation, 502
- Asynchronous chat room communication,
- crisis management, 157–158
- Asynchronous coordination, 397–398
- Asynchronous design
- AMI (asynchronous method invocation), 395–396
  - architectural principles, 202, 205–206
  - asynchronous coordination, 397–398
  - asynchronous scaling, *vs.* synchronous, 396–398
  - asynchronous systems, example, 398–401
  - avoiding session storage, 403–404, 405
  - centralizing session storage, 404, 405
  - Christmas tree lights analogy, 400–401
  - cyclomatic complexity, 400
  - decentralizing session storage, 404, 405
  - declarative programming, 402
  - defining state, 401
  - dining philosophers problem, 394
  - functional programming, 402
  - imperative programming, 402
  - logical programming, 402
  - Mealy machines, 401–402
  - Moore machines, 401–402
  - multiplicative effect of failures, 400–401
  - mutex synchronization, 394
  - mutual exclusion synchronization, 394

- Asynchronous design (*continued*)
    - object-oriented programming, 402
    - procedural programming, 402
    - session environments, saving, 403–404
    - state, saving, 403–404
    - structured programming, 402
    - synchronization process, description, 393–394
    - synchronous calls, example, 395
    - synchronous calls *vs.* asynchronous, 395–401
    - synchronous systems, scaling issues, 398–401
  - Asynchronous method invocation (AMI), 395–396
  - Asynchronous scaling, *vs.* synchronous, 396–398
  - Asynchronous systems, example, 398–401
  - Atomicity, Consistency, Isolation, Durability (ACID) database properties, 383
  - Autonomic Computing Manifesto (IBM), 427–428
  - Availability
    - fault isolation, 312–315
    - TAA effects, 306
    - TAD effects, 306
  - Availability, calculating
    - customer complaints, 515–516
    - hardware uptime, 514–515
    - overview, 513–514
    - portion of site down, 516–517
    - third-party monitoring, 517–518
    - traffic graphs, 518–519
  - Axes, AKF Scale Cube. *See also* X-axis; Y-axis; Z-axis.
    - initial point, 327
    - point of intersection, 327
  - Axes of scale, architectural principles, 203, 207–208
- B**
- Back office grids, 464
  - Back office IT systems
    - AKF Scale Cube for applications, 352–353
    - AKF Scale Cube for databases, 372–373
  - Backbones, clouds, 435–436
  - Backup storage, cost of, 413
  - Backward compatibility, architectural principles, 201
  - Barrier conditions
    - agile development, 275–277
    - Agile Manifesto, 275–277
    - ARBs, 274
    - code reviews, 274
    - cowboy coding, 277
    - definition, 274
    - examples, 274–275
    - hybrid development models, 278
    - performance testing, 275
    - production monitoring and measurement, 275
    - to rollback, 281
    - waterfall development, 277–278
  - Baseline, establishing for stress testing, 265
  - Batch cache refresh, 379–381
  - Behavior (team), evaluating, 97
  - Benchmarks, performance testing, 258–259
  - Berkeley, use of cloud computing, 447
  - Board constituency, ARB, 223–225
  - Books and publications
    - “Above the Clouds: A Berkeley View . . .”, 447
    - Autonomic Computing Manifesto (IBM), 427–428
    - Good to Great*, 71
    - The Grid: Blueprint for a New Computing Infrastructure*, 428
    - The Mythical Man Month*, 429
    - Mythical Man Month . . .*, 49
    - Resonant Leadership*, 68
  - Bottlenecks, identifying for stress testing, 266
  - Boundary conflicts, 15
  - Boyatzis, Richard, 68
  - Brooks, Frederick P., 49, 429
  - Brooks’ Law, 429
  - Buffers, *vs.* caches, 378
  - Build grids, 462–463
  - Build steps, check list, 463
  - Building components *vs.* buying architectural principles, 203
  - being the best, 236
  - case study, 240–242

- competitive components, identifying, 239
  - cost effectiveness, 239–240
  - cost-focused approach, 234–235, 237–240
  - creating strategic competitive
    - differentiation, 238
  - decision checklist, 238–240
  - effects on scalability, 233–234
  - “not built here” phenomenon, 236–237
  - ownership, pros and cons, 238–239
  - strategy-focused approach, 235–236, 237–240
  - TAA conflicts, 305
  - TAD, checklist, 304
  - TAD conflicts, 305
  - Bureaucracy, effect on business processes, 130
  - Business acumen, balancing with technical, 28–29
  - Business case for scale. *See also* Corporate mindset, changing.
    - Amazon outage example, 116
    - customer acquisition costs, 116
    - downtime costs, 114–117
    - intraorganizational costs, 116–117
    - lost customers, 116
  - Business change calendar, 175
  - Business growth rate, determining
    - headroom, 186–187
  - Business metrics, monitoring, 476–477
  - Business processes. *See also* Standards; *specific processes.*
    - bureaucracy, 130
    - capability levels, 125, 126–128
    - CMM (Capability Maturity Model), 124–125
    - CMMI (Capability Maturity Model Integrated), 124–125
    - complexity, managing, 128–130
    - continuous process improvement, 178–179
    - culture clash, 130
    - definition, 122
    - effects on creativity, 123–124
    - excessive mundane work, 127
    - feedback from teams, 131
    - maturity levels, 125, 126–127
    - need for improvement, warning signs, 127
    - periodic maintenance, 131
    - problem causes, 130
    - problems, avoiding, 131
    - purpose of, 122–124
    - repeatability, 126–128
    - repetitive management, same task, 127
    - resolving procedural uncertainty, 123
    - rigor, 126–128
    - same task, multiple procedures, 127
    - standardization, 123
  - Business unit owners, roles and responsibilities, 27
  - Buy when non core, architectural principles, 203
- ## C
- Cache ratio, 379
  - Cache-hits, 379
  - Cache-miss, refreshing caches, 379–381
  - Cache-misses, 379
  - Caches
    - adding data, 382
    - batch cache refresh, 379–381
    - vs.* buffers, 378
    - cache ratio, 379
    - cache-hits, 379
    - cache-misses, 379
    - CDNs (content delivery networks), 389–390
    - datum, 378
    - definition, 378
    - get method, 382
    - hit ratio, 379
    - marshalling, 381
    - memcached, 382
    - object, 381–384
    - reading from, 379
    - retrieving data, 382
    - structure of, 378
    - tags, 378
    - unmarshalling, 381
    - updating. *See* Caches, refreshing.
  - Caches, application
    - controlling with HTML meta tags, 386–387
    - overview, 384
    - proxy caches, 384–385
    - reverse proxy caches, 386–387
    - software for, 388–389

- Caches, refreshing
  - LRU (least recently used) algorithm, 379–381
  - MRU (most recently used) algorithm, 379–381
  - overview, 379–381
  - replace method, 382
  - upon cache-miss, 379–381
- Caches, writing to
  - dirty data, 381
  - set method, 382
  - write-back method, 381
  - write-through policy, 381
- Caching algorithms, 379–380
- Calculation formula, determining headroom, 188–189
- Capability levels, business processes, 125, 126–128
- Capability Maturity Model (CMM), 124–125
- Capability Maturity Model Integrated (CMMI), 124–125
- Capacity, system. *See* Headroom.
- Capacity planners, roles and responsibilities, 35
- Capacity planning
  - calculations, 521–526
  - grids, utilization of unused capacity, 457
  - roles and responsibilities, 32
  - x-axis splits, databases, 361
- Case studies
  - building components *vs.* buying, 240–242
  - change, air traffic control, 170
  - crisis management, 152, 505–506
  - eBay, 152, 505–506
  - Quigo, 506–507
  - ShareThis, 507–508
- Causal roadmap to success, 84–86
- CDNs (content delivery networks), 389–390
- CEO (Chief Executive Officer)
  - executive interrogations, 25
  - outside help, 26
  - roles and responsibilities, 25–26
  - scalability proficiency, 26
  - scalability roles and responsibilities, 25–26
  - seeking consistent explanations, 25–26
- CFO (Chief Financial Officer), roles and responsibilities, 26–27
- Change
  - definition, 166–167
  - identifying, 168–170
  - in individuals, effecting through leadership, 68
  - logging, 168–170
  - in small companies, 167–168
- Change Approval Board, 178
- Change Control Meeting, 178
- Change management. *See also* ARB (Architecture Review Board); JAD (Joint Architecture Design).
  - air traffic control case study, 170
  - approval, 174
  - benefits of proposed change, 175–176
  - business change calendar, 175
  - Change Approval Board, 178
  - Change Control Meeting, 178
  - change proposal, 172–174
  - checklist, 179
  - continuous process improvement, 178–179
  - cumulative changes and risk management, 253–254
  - effectiveness review, 177–178
  - expected results, 173
  - goal of, 171
  - implementation, 176
  - ITIL (Information Technology Infrastructure Library), 171
  - key metrics, 177–178
  - logging, 176
  - overview, 170–171
  - relationship to other systems, 173–174
  - request for change, 172
  - required information, 172–174
  - risk assessment, 173, 175, 253–254
  - scheduling, 174–176
  - target system, identifying, 172–173
  - undoing changes. *See* Rollback.
  - validation, 176–177
- Change proposal, 172–174
- Chat room communication, crisis management, 157–158
- Check lists and guidelines
  - adopting scalability, 501–504
  - ARB (Architecture Review Board), 229–230

- benefits of cloud computing, 442
- build steps, 463
- building components *vs.* buying, 238–240
- change management, 179
- cloud characteristics, 433
- cloud computing, cons, 446–447
- communications improvement, 107–108
- costs of data, 414
- deciding to use cloud computing, 452
- defeating the corporate mindset, 110. *See also specific guidelines.*
- developing architectural principles, 199–200
- failure to catch problems, 470–471
- fault isolation, 321
- headroom, determining, 192
- improving communications, 107–108
- incident management, 134
- JAD sessions, 216–217
- making tradeoff decisions, 294
- measuring risk, 252
- monitoring issues, 478
- multiple live data center considerations, 497
- organizational design, 12
- performance testing, steps in, 263. *See also specific steps.*
- recommended architectural principles, 208
- rollback requirements, 280
- stress testing, 268–269
- team size, optimizing, 54
- Check lists and guidelines, grid computing cons, 460
  - deciding to use grid computing, 467
  - pros, 458
- Chief Executive Officer (CEO). *See* CEO (Chief Executive Officer).
- Chief Financial Officer (CFO), roles and responsibilities, 26–27
- Chief scalability officer. *See* CEO (Chief Executive Officer).
- Chief Technology Officer (CTO), roles and responsibilities, 27–29
- Christmas tree lights analogy, 400–401
- CIO (Chief Information Officer), roles and responsibilities, 27–29
- Clarity, roles and responsibilities, 22–23
- Classification, incident management, 138
- Cloning
  - data, no bias, 358–362
  - services, no bias, 341–342
  - services and data, no bias, 328–330
- Closed incidents, 141
- Closed problems, 141
- Closure, incidents, 138
- Cloud computing. *See also* Grid computing.
  - deciding to use, 450–453
  - environmental requirements, 448–449
  - implementation requirements, 448–450
  - skill sets required, 449–450
- Cloud computing, cons
  - “Above the Clouds: A Berkeley View . . .”, 447
  - control, 444
  - IP address limitations, 444–445
  - limitations, 444–445
  - load balancing, 444–445
  - performance, 445–446
  - portability, 443
  - security, 443
  - summary of, 446–447
  - third-party software certification, 444–445
  - top ten obstacles, 447
- Cloud computing, history of
  - Artificial Intelligence, 427–428
  - dot com bubble, 427
  - IaaS (Infrastructure as a Service), 427–428
  - IBM, Autonomic Computing Manifesto, 427–428
  - PaaS (Platform-as-a-Service), 427–428
  - SaaS (Software as a Service), 427–428
  - XaaS (Everything as a Service), 427–428
- Cloud computing, pros
  - cost, 440–441
  - flexibility, 442
  - speed, 441–442
  - summary of, 442
- Clouds
  - backbones, 435–436
  - definition, 426
  - vs.* grids, 434–436
  - hypervisors, 433

- Clouds (*continued*)
  - public *vs.* private, 426
  - service providers, 435–436
  - types of, 435–436
  - virtualization software, 435–436
- Clouds, characteristics of
  - multiple tenants, 432–433, 434
  - pay by usage, 431, 434
  - in private clouds, 434
  - scale on demand, 431–432, 434
  - summary of, 433
  - virtualization, 433, 434
- CMM (Capability Maturity Model), 124–125
- CMMI (Capability Maturity Model Integrated), 124–125
- COBIT (Control Objectives and related Technology), 133–134
- Code complexity, AKF Scale Cube for applications, 346
- Code reviews, as barrier conditions, 274
- Codebase increases, scaling for, 343–344
- Collins, Jim, 71
- Commoditization over time, 301–302
- Commodity hardware, architectural principles, 203
- Communication breakdown
  - from the business end, 106–107
  - destructive interference, 105–106, 212
  - educational mismatch, 106
  - the experiential chasm, 105–108
  - between management and technical teams, 105–108
  - matrix organization teams, 59
  - team size, warning sign, 50–51
  - from the technical end, 107–108
- Communications
  - improvement check lists and guidelines, 107–108
  - organizational design, influences on, 44
  - team size, 48
- Communications and control, crisis management, 157–158
- Comparing
  - pros and cons, making tradeoff decisions, 291–292
  - real *vs.* ideal situations, 504
- Compassion, leadership, 68
- Competitive components, identifying, 239
- Complexity
  - business processes, managing, 128–130
  - grid drawback, 459–460
- Concepts *vs.* rules, AKF Scale Cube, 325–326
- Conditional approval by the ARB, 226, 229
- Configuration management
  - AKF Scale Cube for applications, 342
  - x-axis splits, databases, 361
- Conflicts, organizational design, 17
- Consensus, JAD (Joint Architecture Design), 218
- Consulted persons, RASCI, 38
- Content delivery networks (CDNs), 389–390
- Control, cloud computing drawback, 444
- Control Objectives and related Technology (COBIT), 133–134
- Corporate mindset. *See also* Roles and responsibilities, executives.
  - by business type, 109–110
- Corporate mindset, changing. *See also* Business case for scale.
  - check list and guidelines, 110–114
  - educating executives, 111–112
  - forming relationships, 111
  - involving executives, 113
  - “scared straight” method, 113–114
  - setting an example, 111
  - speaking in business terms, 112–113
  - technology ride-alongs, 112
  - using the RASCI model, 112
- Cost
  - cloud computing benefit, 440–441
  - of data. *See* Data, cost of.
  - fault isolation, 316–317
  - grid benefit, 457–458
  - rollback, 281–282
  - of scale, goals, 99
  - TAD (technology agnostic design), 301–302
  - tradeoffs in business, 286
- Cost, data centers
  - HVAC power, 491
  - operations, 491
  - planning for, 483–485
  - servers, 491

- Cost, of scalability failure
    - customer acquisition costs, 116
    - downtime costs, 114–117
    - intraorganizational costs, 116–117
    - lost customers, 116
  - Cost, scaling applications. *See also* AKF
    - Scale Cube for applications, cost.
    - x-axis splits, 341–342
    - y-axis splits, 344
    - z-axis splits, 345–347
  - Cost, splitting databases
    - x-axis splits, 360, 361–362
    - y-axis splits, 363, 364
    - z-axis splits, 366, 367
  - Cost centers, 109
  - Cost effectiveness, building components *vs.* buying, 239–240
  - Cost-focused approach, building components *vs.* buying, 234–235, 237–240
  - Cost/speed/quality/scope triangle, 285–288
  - Cost-Value Data Dilemma, 414–415
  - Cowboy coding, 277
  - Creativity, effects of business processes, 123–124
  - Credibility, leadership, 70, 73
  - Crises. *See also* Incidents; Issues; Problems.
    - vs.* common incidents, 150–151
    - crisis threshold, 150
    - definition, 149–150
    - eBay case study, 152
    - effects on a company, 151
    - failure to correct, consequences of, 151
    - potential benefits, 151
    - prioritizing, 150
  - Crises, management. *See also* Incident management; Postmortem review; Problem management.
    - asynchronous chat room communication, 157–158
    - case studies, 505–506
    - communications and control, 157–158
    - customer apologies, 162–163
    - documenting activities, 154
    - engineering leads, role of, 156
    - escalations, 160
    - follow up communication, 162
    - individual contributors, role of, 157
    - master postmortem, 162
    - need for coordination, 153
    - overview, 152–153
    - postmortems, 161–162
    - problem manager, role of, 153–155
    - RASCI framework, 160
    - status communications, 160–161
    - synchronous voice communications, 157–158
    - team managers, role of, 155–156
    - war room, 158–159
  - Crisis threshold, 150
  - Criteria, performance testing, 258–259
  - Criticality of services, ranking for stress testing, 265
  - CTO (Chief Technology Officer), roles and responsibilities, 27–29
  - Cultural aspects, TAD (technology agnostic design), 307–308
  - Cultural interviews, managing teams, 94–98
  - Culture clash, effect on business processes, 130
  - Customers
    - acquisition, costs of scalability failure, 116
    - apologies, 162–163
    - base growth, scaling for. *See* Z-axis splits.
    - complaints, calculating availability, 515–516
    - experience monitoring, 476
    - splitting work responsibility by, 333–334
  - Cyclomatic complexity, 400
- ## D
- Daily Incident Meeting, 141–142
  - Daily incident review, 141–143
  - Data
    - collection, for stress testing, 267
    - consistency, database x-axis splits, 360–362
    - currency, database x-axis splits, 362
    - handling large amounts of. *See* Caching.
    - increases, scaling for, 342
    - separating into schemas, 362–365
    - size *vs.* problem specificity, 475
  - Data, cost of
    - backup storage, 413
    - cost justifying storage, 417–419
    - Cost-Value Data Dilemma, 414–415

- Data, cost of (*continued*)
    - ETL (Extract, Transform, and Load), 420
    - handling large amounts of, 420–423
    - initial cost of storage, 412–413
    - MAID (Massive Array of Idle Disks), 412
    - mapping data, 420–423
    - MapReduce program, 420–423
    - option value of data, 414–415, 416
    - peak utilization time, 413
    - reducing data, 420–423
    - shareholder value, diluting, 415
    - strategic advantage value of data, 415, 416–417
    - summary of, 414
    - tiered storage solutions, 417–419
    - transforming for storage, 419–420
    - unprofitable data, eliminating, 415
  - Data center planning
    - choosing a location, 485–488
    - constraints, 483–485
    - costs, 483–485
    - disaster recovery, 496–497
    - elevation, importance of, 486
    - humidity factor, 486
    - HVAC considerations, 486
    - incremental growth, 488–490
    - multiple active centers, 496–497
    - power requirements, 484–485
    - quality of service, 486–487
    - rack units, 484
    - risk profile, 486–487
  - Data center planning, Three Magic Rules of
    - Threes
      - HVAC power costs, 491
      - number of data centers, 492–495
      - number of servers, 491–492
      - operational costs, 491
      - overview, 490
      - server power costs, 491
      - servers, 491
  - Data warehouse grids, 463–464
  - Database administrators, roles and responsibilities, 34–35
  - Database increases, scaling for, 343–344
  - Databases
    - ACID (Atomicity, Consistency, Isolation, Durability) database properties, 383
    - scaling. *See* AKF Scale Cube for databases.
  - Datum, caches, 378
  - Decision making, leadership, 73
  - Decision matrix method, making tradeoff decisions, 292–295
  - Declaration of Independence, 76–77
  - Declarative programming, 402
  - Delegation, importance of, 24
  - Designing scalable systems. *See* Architectural principles; TAD (technology agnostic design).
  - Designing to be disabled
    - architectural principles, 201
    - markdown functionality, 282–283
  - Designing to be monitored, 470–471
  - Destructive interference
    - communication breakdown, 105–106
    - in dysfunctional organizations, 212
  - Detection, incident management, 136–138
  - Diagnosis, incident management, 138
  - Dining philosophers problem, 394
  - Dirty cache data, 381
  - Disaster recovery, data center planning, 496–497
  - Distributed object caches, AKF Scale Cube for databases, 360
  - Documentation, crisis management activities, 154
  - Dot com bubble, 427
  - Downtime costs, of scalability failure, 114–117
  - DRIER (Detect, Report, Investigate, Escalate, Resolve) process, incident management, 138–139, 146–147
  - Dunning, David, 67
  - Dunning-Kruger effect, 67–68
  - Dysfunctional organizations, 211–213
- ## E
- EBay, crisis management case studies, 152, 505–506
  - Ecommerce
    - AKF Scale Cube for applications, 350–351
    - AKF Scale Cube for databases, 370–372
  - Educating executives, 111–112
  - Educational mismatch, communication breakdown, 106

- Efficiency, influences of organizational design, 44
  - Ego, role in leadership, 71
  - Elevation, data centers, 486
  - Employee reviews, leadership, 69
  - Employees. *See* Roles and responsibilities; Staffing; *specific roles*.
  - Empowerment, JAD (Joint Architecture Design), 218
  - Engineering, roles and responsibilities, 30
  - Engineering leads, role in crisis management, 156
  - Engineering teams
    - causal roadmap to success, 85
    - productivity and efficiency, 100
  - Entry criteria
    - ARB (Architecture Review Board), 228–229
    - JAD (Joint Architecture Design), 217–218
  - Environment, creating for
    - performance testing, 259
    - stress testing, 266
  - ERP (enterprise resource planning)
    - AKF Scale Cube for applications, 351–352
    - AKF Scale Cube for databases, 372
  - Error chains, 1
  - Escalations, crisis management, 160
  - Ethical behavior
    - leadership, 70, 73
    - management, 90
  - ETL (Extract, Transform, and Load), 420
  - Everything as a Service (XaaS), 427–428
  - Examples. *See* Case studies.
  - Executives
    - making the business case for scale. *See* Business case for scale; Corporate mindset, changing.
    - roles and responsibilities. *See* Roles and responsibilities, executives.
  - Exit criteria
    - ARB (Architecture Review Board), 228–229
    - JAD (Joint Architecture Design), 218–219
  - Experiential chasm, communication
    - breakdown, 105–108
- F**
- Failure
    - to catch problems, 470–471
    - costs of. *See* Cost, of scalability failure.
    - detection, 249
    - likelihood, 249
    - multiplicative effect of, 400–401
    - risk management, 249
    - severity, 249
    - stress testing, 265
  - Failure Mode and Effects Analysis (FMEA), 249–251
  - Fault isolation
    - design checklist, 321
    - markdown functionality, 313–314
    - poddings, definition, 310
    - pods, definition, 310, 311
    - pools, definition, 311
    - sharding, definition, 311
    - shards, definition, 311
    - slivers, definition, 311
    - terminology, 310–311
    - testing, 321–322
    - x-axis splits, 343
    - y-axis splits, 345
    - z-axis splits, 346
  - Fault isolation, approaches to
    - along natural barriers, 320–321
    - by the biggest sources of incidents, 320
    - design checklist, 321
    - isolating the money-maker, 320
  - Fault isolation, benefits of
    - availability, 312–315
    - cost, 316–317
    - effects on revenue, 317
    - incident detection, 315
    - incident resolution, 315
    - scalability, 315
    - time to market, 315–316
  - Fault isolation, design principles
    - design checklist, 321
    - nothing crosses a swim lane boundary, 319
    - nothing is shared, 318
    - transactions occur along swim lanes, 319
  - Fault isolation, swim lanes
    - along natural barriers, 320–321

Fault isolation, swim lanes (*continued*)  
 by the biggest sources of incidents, 320  
 by customer boundaries, 312–313  
 definition, 310, 311  
 isolating the money-maker, 320  
 by service boundaries, 313–314

Feature selection, ARB (Architecture Review Board), 228

Feature significance, JAD (Joint Architecture Design), 217

Feedback from teams, business processes, 131

Feeding teams, 98

Final design, documenting in JAD (Joint Architecture Design), 218–219

Fisher, Mike, 506, 507–508

Flexibility, cloud computing benefit, 442

FMEA (Failure Mode and Effects Analysis), 249–251

Follow up communication, crisis management, 162

Formal training, leadership, 68

Forward proxy caches. *See* Proxy caches.

Foster, Ian, 428

Functional organization. *See* Team structure, functional organization.

Functional programming, 402

## G

Gates, Bill, 45

General managers, roles and responsibilities, 27

Get method, 382

Globus Toolkit, 429

Goal trees, 101–102, 196

Goals for achieving scalability, 503–504. *See also* Leadership, goals; Management, goals.

*Good to Great*, 71

Google, MapReduce program, 420–423

Google Inc., cloud computing, 430

*The Grid: Blueprint for a New Computing Infrastructure*, 428

Grid computing. *See also* Cloud computing.  
 deciding to use, 465–467  
 definition, 428  
*The Grid: Blueprint for a New Computing Infrastructure*, 428

middleware providers, 429  
 public *vs.* private networks, 429

Grid computing, uses for  
 back office grid, 464  
 build grid, 462–463  
 data warehouse grid, 463–464  
 MapReduce, 464  
 production grid, 461–462  
 SETI@home project, 429

Grids, cons  
 complexity, 459–460  
 monolithic applications, 459  
 not shared simultaneously, 459  
 summary of, 460

Grids, pros  
 cost, 457–458  
 high computational rates, 456–457  
 shared infrastructure, 457  
 summary of, 458  
 utilization of unused capacity, 457

Grids, *vs.* clouds, 434–436

Guidelines. *See* Check lists and guidelines.

Gut feel method  
 making tradeoff decisions, 291  
 risk management, 245–246

## H

Hardware. *See also specific hardware.*  
 architectural principles, 203  
 headroom, determining, 185  
 uptime, calculating availability, 514–515

Headroom, definition, 183

Headroom, determining  
 actual usage, determining, 186  
 for annual budget, 184  
 business growth rate, determining, 186–187  
 calculation checklist, 192  
 calculation formula, 188–189  
 excessive swapping, 190  
 hardware implications, 185  
 for hiring plans, 184

Ideal Usage Percentage, 189–191

infrastructure features, 187–188

negative headroom, 188–189

positive headroom, 188–189

prioritizing projects, 185

- process description, 185–189
  - purpose of, 184–185
  - scalability projects, 187–188
  - seasonality effect, 187
  - statistical load averages, 191
  - system components, identifying, 186
  - thrashing, 190
  - Hewlett-Packard Company, cloud
    - computing, 430
  - High computational rates, grid benefit, 456–457
  - Hiring decisions, managing teams, 95
  - Hiring plans, determining headroom, 184
  - Hit ratio, caches, 379
  - Hope, leadership, 68
  - HTML meta tags, controlling application
    - caches, 386–387
  - Human factor, in risk management, 254
  - Human resources ERP systems
    - AKF Scale Cube for applications, 351–352
    - AKF Scale Cube for databases, 372
  - Humidity, data centers, 486
  - HVAC, data center planning, 486, 491
  - Hybrid development models, barrier
    - conditions, 278
  - Hypervisors, clouds, 433
- I**
- IaaS (Infrastructure as a Service), 427–428
  - IBM, Autonomic Computing Manifesto, 427–428
  - Ideal Usage Percentage, 189–191
  - Identified problems, 141
  - Identifying, architectural principles, 199
  - Image storage feature, ARB (Architecture Review Board) example, 227–228
  - Image storage feature, JAD example, 215–216
  - Imperative programming, 402
  - Implementation *vs.* architecture, 300
  - Incident detection, fault isolation, 315
  - Incident management. *See also* Postmortem review; Problem management.
    - check list, 134
    - classification and initial support, 138
    - components of, 136–139
    - conflicts with problem management, 140
    - Daily Incident Meeting, 141–142
    - detection and recording, 136–138
    - DRIER (Detect, Report, Investigate, Escalate, Resolve) process, 138–139, 146–147
    - essential activities, 136–139
    - follow-up ownership, 138
    - incident closure, 138
    - investigation and diagnosis, 138
    - monitoring incidents, 137
    - resolution and recovery, 138
  - Incident management, incident review
    - after action review meeting. *See* Postmortem review.
    - daily, 141–143
    - developing a timeline, 144
    - quarterly, 143, 146, 147
  - Incident managers, roles and responsibilities, 34
  - Incident resolution, fault isolation, 315
  - Incidents. *See also* Crises; Issues; Problems.
    - closed, definition, 141
    - closure, 138
    - vs.* crises, 150–151
    - definition, 134–135
    - life cycle, 140–141, 146
    - monitoring, 137
    - open, definition, 141
    - relationship to problems, 139
    - resolved, definition, 141
    - reviewing. *See* Incident management, incident review.
    - selecting for postmortem, 144
  - Individual contributors, roles and responsibilities. *See* Roles and responsibilities, individual contributors.
  - Individual focus *vs.* team focus, leadership, 71
  - Information Technology Infrastructure Library (ITIL), 133, 171
  - Informed persons, RASCI, 38
  - Infrastructure, roles and responsibilities, 31
  - Infrastructure as a Service (IaaS), 427–428
  - Infrastructure engineers, roles and responsibilities, 34–35
  - Infrastructure features, determining headroom, 187–188

Initial support, incident management, 138  
 Innovation, leadership, 66  
 Interactivity of system services, stress testing, 265  
 Interviewing candidates, managing teams, 95  
 Intraorganizational costs, of scalability failure, 116–117  
 Investigation, incident management, 138  
 IP address limitations, cloud computing drawback, 444–445  
 Issue Phase, postmortem review, 145  
 Issues. *See also* Crises; Incidents; Problems.  
   identification, 31, 145  
   management, roles and responsibilities, 31  
   in the postmortem review, 145  
   roles and responsibilities, 31  
 ITIL (Information Technology Infrastructure Library), 133, 171

## J

JAD (Joint Architecture Design). *See also* ARB (Architecture Review Board).  
   applicable development methodologies, 214  
   ARB approval, 219  
   architectural principles, 218  
   consensus, 218  
   description, 214–215  
   in dysfunctional organizations, 211–213  
   empowerment, 218  
   entry criteria, 217–218  
   established teams, 218  
   exit criteria, 218–219  
   feature significance, 217  
   final design, documenting, 218–219  
   image storage feature, example, 215–216  
   product requirements, 218  
   scaling cross functionally, 214–215  
   session checklist, 216–217  
   structure of, 214–215  
   tradeoffs, documenting, 218  
 Justifying, data storage costs, 417–419

## K

Keeven, Tom, 506  
 Kesselman, Carl, 428

Key services, identifying for stress testing, 265  
 Kishore, Nanda, 507–508  
 Klieber's Law of metabolism, 260  
 Kruger, Justin, 67

## L

Last-mile monitoring, 476  
 Leadership. *See also* Management;  
   Organizational design; Staffing for scalability.  
   360-degree reviews, 69  
   abuse of, 70  
   alignment with shareholder values, 74–75  
   born *vs.* made, 65  
   causal roadmap to success, 84–86  
   characteristics of, 66–67  
   compassion, 68  
   credibility, 70, 73  
   decision making, 73  
   definition, 64  
   destructive behavior, 73  
   Dunning-Kruger effect, 67–68  
   effecting change in individuals, 68  
   ego, role of, 71  
   employee reviews, 69  
   ethical behavior, 70, 73  
   formal training, 68  
   *Good to Great*, 71  
   hope, 68  
   individual focus *vs.* team focus, 71  
   innovation, 66  
   leading from the front, 69–70  
   *vs.* management, 64–65  
   mindfulness, 68  
   perception by others, 66–67  
   perceptions of others, 73  
   perseverance, 66  
   pulling activities, 18, 64  
   *Resonant Leadership*, 68  
   role of, 19  
   self-assessment, 67–69  
   SWOT (strength, weaknesses, opportunities, threats) analysis, 82  
   team empowerment, and scalability, 74  
 Leadership, goals. *See also* Causal roadmap to success.  
   definition, 79

- developing, example of, 81–84
  - SMART (Specific, Measurable, Attainable, Realistic, Timely) goals, 80–81, 83
  - Leadership, mission. *See also* Causal roadmap to success.
    - definition, 78
    - development, example of, 81–84
    - examples of, 78–79
    - Mission First, 72
    - Mission First, Me Always, 72
    - Mission First, People Always, 72
    - mission statements, 78–79, 82
    - People Always, 72
    - people *vs.* mission, 72
  - Leadership, vision. *See also* Causal roadmap to success.
    - components of, 78
    - definition, 75
    - developing, example of, 81–84
    - examples of, 76–77
    - vision statements, 77–78
  - Leading from the front, 28, 69–70
  - Life cycle
    - incidents, 140–141, 146
    - problems, 140–141
  - Load
    - averages, statistical, 191
    - balancing, cloud computing drawback, 444–445
    - calculating, 527–533
    - simulating for stress testing, 267
    - testing, 258
  - Logging changes, 168–170, 176
  - Logical programming, 402
  - Longhorn. *See* Microsoft Vista.
  - Lost customers, of scalability failure, 116
  - LRU (least recently used) algorithm, 379–381
- M**
- Madoff, Bernie, 73
  - MAID (Massive Array of Idle Disks), 412
  - Maintenance, business processes, 131
  - Management. *See also* Leadership; Organizational design; Staffing for scalability.
    - definition, 90
    - desirable characteristics, 90–91
    - ethical behavior, 90
    - good *vs.* great, 91
    - vs.* leadership, 64–65
    - paving the path to success, 102–103
    - projects, 91–92
    - pushing activities, 18
    - role of, 18–19
    - tasks, 91–92
  - Management, goals
    - cost of scale, 99
    - deciding what to measure, 99
    - engineering productivity and efficiency, 100
    - goal trees, 101–102
    - mapping, 101–102
    - metrics for goal evaluation, 98–101
    - quality, 100–101
    - relative response time, 99
    - system response time, 99
    - user-perceived response time, 99
  - Management, metrics
    - cost of scale, 99
    - deciding what to measure, 99
    - engineering productivity and efficiency, 100
    - for goal evaluation, 98–101
    - quality, 100–101
    - relative response time, 99
    - system response time, 99
    - user-perceived response time, 99
  - Management, of teams
    - cultural interviews, 94–98
    - evaluating behavior and performance, 97
    - feeding, 98
    - hiring decisions, 95
    - interviewing candidates, 95
    - seeding, 98
    - team building, 93–94
    - team upgrading, 94–98
    - terminating members, 96
    - weeding, 98
  - Management style, 18
  - Mapping, goals, 101–102
  - Mapping data, 420–423
  - MapReduce program, 420–423, 464
  - Markdown functionality, 282–283, 313–314.
    - See also* Designing to be disabled.

- Marshalling caches, 381
  - Massive Array of Idle Disks (MAID), 412
  - Master postmortem, 162
  - Matrix organization. *See* Team structure, matrix organization.
  - Mature technologies, architectural principles, 202
  - Maturity levels, business processes, 125, 126–127
  - McCarthy, John, 427
  - McKee, Annie, 68
  - Mealy machines, 401–402
  - Measurable architectural principles, 198. *See also* Metrics.
  - Meetings, of the ARB, 225–227
  - Memcached, 382
  - Metrics
    - change management, 177–178
    - cost of scale, 99
    - deciding what to measure, 99
    - engineering productivity and efficiency, 100
    - goal evaluation, 98–101
    - for goal evaluation, 98–101
    - organizational design for, 12–13
    - quality, 100–101
    - relative response time, 99
    - system response time, 99
    - user-perceived response time, 99
  - Micromanagement of teams, 51
  - Microsoft Corporation, cloud computing, 430
  - Microsoft Vista, standards failure, 45–46
  - Mindfulness, leadership, 68
  - Missing data, identifying in the postmortem review, 145
  - Mission
    - leadership styles. *See* Leadership, mission. *vs.* people, 72
  - Mission First, Me Always leadership style, 72
  - Mission First, People Always leadership style, 72
  - Mission First leadership style, 72
  - Mission statements, 78–79, 82
  - Mistakes, identifying in the postmortem review, 145
  - Monitoring
    - architectural principles, 202, 204
    - in business processes, 480–481
    - calculating availability, third-party services, 517–518
    - designing to be monitored, 470–471
    - failure to catch problems, 470–471
    - incidents, 137
    - issues, 478
    - maturing, 137
  - Monitoring, framework for
    - application monitoring, 477–478
    - business metrics, 476–477
    - customer experience monitoring, 476
    - data size *vs.* problem specificity, 475
    - identifying a problem, 473
    - identifying problem cause, 474–476
    - isolating a problem, 473–474
    - last-mile monitoring, 476
    - overview, 472–473
    - systems monitoring, 477
    - user experience, 476–477
  - Monitoring, measuring
    - determining what to monitor, 478–480
    - identifying a problem, 479
    - identifying problem cause, 479
    - isolating a problem, 479–480
  - Monitors, identifying for stress testing, 267
  - Monolithic applications, grid drawback, 459
  - Moore machines, 401–402
  - Morale issues, team warning sign, 50–51
  - Most-used scenarios, performance testing, 261
  - Most-visible scenarios, performance testing, 261
  - MRU (most recently used) algorithm, 379–381
  - Multiple live sites, architectural principles, 202, 205
  - Multiple tenants, clouds, 432–433, 434
  - Multiplicative effect of failures, 400–401
  - Mutex synchronization, 394
  - Mutual exclusion synchronization, 394
  - The Mythical Man Month . . .*, 49, 429
- ## N
- N+1 design, architectural principles, 200–201
  - Needs of the business, team size, 49
  - Negative headroom, 188–189
  - Negative testing, stress testing, 264–265, 265

Network engineers, roles and responsibilities, 34–35  
 “Not built here” phenomenon, 236–237  
 Not shared simultaneously, grid drawback, 459  
 Noun perspective, y-axis database splits, 363

## O

Object caches, 381–384  
 Object-oriented programming, 402  
 Open incidents, 141  
 Open problems, 141  
 Operations teams, causal roadmap to success, 85–86  
 Operators, roles and responsibilities, 34  
 Optimal range, team size, 46–47  
 Option value of data, 414–415, 416  
 Organizational design. *See also* Leadership; Management; Staffing for scalability; Teams.  
   adding/removing people, 12, 13  
   aligning with stakeholder interests, 13  
   boundary conflicts, 15  
   cost centers, 109  
   drawbacks, 17  
   efficient work flow, 15–17  
   helpful questions, 12  
   metrics, 12–13  
   potential conflicts, 17  
   profit centers, 109  
   support services business, 109–110  
   team growth *vs.* individual productivity, 13–15  
   technology-as-product business, 109–110  
   understanding, from the outside, 13  
 Organizational design, influences on scalability  
   communications, 44  
   efficiency, 44  
   ownership, 46  
   quality, 45  
   standards, 44–45  
 Organizational roles and responsibilities. *See* Roles and responsibilities, organizational.  
 Overall risk, 253–254  
 Overlapping roles and responsibilities, 40  
 Overwork, team warning sign, 52

## Ownership

architectural principles, 199–200  
 assigning to postmortem tasks, 145  
 building components *vs.* buying, 238–239  
 incident follow-up activities, 138  
 organizational design, influences on, 46

## P

PaaS (Platform as a Service), 427–428  
 Pareto, Vilfredo, 260  
 Pareto Distribution, performance testing, 260  
 Pay by usage, clouds, 431, 434  
 Peak utilization time, data cost, 413  
 People. *See* Roles and responsibilities; Staffing; *specific roles.*  
 People Always leadership style, 72  
 People management. *See* Management, of teams.  
 People *vs.* mission, 72  
 Perception by others, leadership, 66–67, 73  
 Performance (product)  
   calculating, 527–533  
   cloud computing drawback, 445–446  
 Performance (product), testing  
   analyzing data, 261–262  
   as barrier conditions, 275  
   benchmarks, 258–259  
   checklist of testing steps, 263. *See also specific steps.*  
   criteria, 258–259  
   defining tests, 260–261  
   definition, 257–258  
   environment, creating, 259  
   executing tests, 261  
   individual components, 261  
   load testing, 258  
   most-used scenarios, 261  
   most-visible scenarios, 261  
   Pareto Distribution, 260  
   repeating tests and analysis, 262–263  
   reporting results to engineers, 262  
   requirements, 258–259  
   for scalability, 270–271  
   stress testing, 265  
 Performance (team), evaluating, 97  
 Perseverance, leadership, 66

- Persistency, AKF Scale Cube for applications, 341
  - P&L owners, roles and responsibilities, 27
  - Platform as a Service (PaaS), 427–428
  - Pledge of Allegiance, 76
  - Poddings, definition, 310
  - Pods, definition, 310, 311
  - Pools, definition, 311
  - Portability, cloud computing drawback, 443
  - Portion of site down, calculating availability, 516–517
  - Positive headroom, 188–189
  - Positive testing, stress testing, 264
  - Postmortem review. *See also* Incident management, incident review.
    - Action Phase, 145
    - assigning owners to tasks, 145
    - attendees, 144
    - creating a task list, 145
    - crisis management, 161–162
    - example, 147
    - input to, 144
    - Issue Phase, 145
    - issues, identifying, 145
    - master postmortem, 162
    - missing data, identifying, 145
    - mistakes, identifying, 145
    - problems, identifying, 145
    - RASCI process, 145
    - reviewing the timeline, 145
    - selecting incidents for review, 144
    - SMART criteria, 145
    - Timeline Phase, 145
  - Power requirements, data centers, 484–485, 491
  - Preamble to the U.S. Constitution
    - mission statement example, 78–79
    - vision statement example, 77
  - Prioritizing
    - crises, 150
    - problems, 142
    - projects, determining headroom, 185
  - Private clouds, characteristics of, 434
  - Private *vs.* public clouds, 426
    - grid networks, 429
  - Problem management. *See also* Incident management; Issues, management; Postmortem.
    - check list, 135
    - components of, 139–140
    - conflicts with incident management, 140
    - objective, 135
  - Problem manager, role in crisis management, 153–155
  - Problems. *See also* Crises; Incidents; Issues.
    - closed, definition, 141
    - definition, 134–135
    - detecting, 473, 479. *See also* Monitoring.
    - failure to detect, 470–471
    - finding the cause, 474–476, 479
    - identified, definition, 141
    - identifying in postmortem, 145
    - isolating, 473–474, 479–480
    - life cycle, 140–141
    - open, definition, 141
    - prioritizing, 142
    - relationship to incidents, 139
    - reviewing, 142
    - specificity, *vs.* data size, 475
  - Procedural programming, 402
  - Process. *See* Business processes.
  - Product requirements, JAD (Joint Architecture Design), 218
  - Production grids, 461–462
  - Production monitoring and measurement, as barrier conditions, 275
  - Production operations, roles and responsibilities, 30–31
  - Productivity, effects of team size, 50–51
  - Profit centers, 109
  - Project management, 91–92
  - Project ownership, matrix organization, 59
  - Project triangle, tradeoffs in business, 285–288
  - Proxy caches, 384–385
  - Proxy servers. *See* Proxy caches.
  - Public *vs.* private clouds, 426
    - grid networks, 429
  - Pulling activities, leadership, 64
- Q**
- QA analysts, roles and responsibilities, 35
  - QA engineers, roles and responsibilities, 35
  - Quality
    - goals, 100–101

- organizational design, influences on, 45
  - tradeoffs in business, 286–287
  - Quality assurance, roles and responsibilities, 31–32
  - Quality of service, data center planning, 486–487
  - Quarterly incident review, 143, 146, 147
  - Quigo, case studies, 506–507
- R**
- Rack units, data center planning, 484
  - Ranking, architectural principles, 199
  - RASCI (Responsible, Approve, Support, Consult, Inform)
    - accountable persons, 38
    - changing the corporate mindset, 112
    - consulted persons, 38
    - crisis management, 160
    - description, 38–41
    - informed persons, 38
    - postmortem review, 145
    - responsible persons, 38
    - sample matrix, 39
    - support persons, 38
  - Reading, from caches, 379
  - Realistic architectural principles, 198
  - Recording incidents, incident management, 136–138
  - Recoverability, stress testing, 265
  - Recovery from incidents, incident management, 138
  - Reducing data, 420–423
  - Reedy, Lynn, 505–506
  - Rejection, by the ARB, 226, 229
  - Removing people, 12. *See also* Staffing.
  - Repeatability, business processes, 126–128
  - Repeating tests and analysis, performance testing, 262–263
  - Replace method, 382
  - Replication delays
    - AKF Scale Cube for databases, 359–360
    - x-axis splits, databases, 359–360, 361
  - Reporting results to engineers, performance testing, 262
  - Request for change, 172
  - Requestor, splitting work responsibility by, 333–334
  - Requirements, performance testing, 258–259
  - Resolution of incidents, incident management, 138
  - Resolved incidents, 141
  - Resonant Leadership*, 68
  - Resources, splitting work responsibility by, 331–332
  - Response time metrics
    - relative response time, 99
    - system response time, 99
    - user-perceived response time, 99
  - Responsibilities. *See* Roles and responsibilities.
  - Responsible, Approve, Support, Consult, Inform (RASCI). *See* RASCI (Responsible, Approve, Support, Consult, Inform).
  - Responsible persons, RASCI, 38
  - Retrieving data, from caches, 382
  - Revenue, effects of fault isolation, 317
  - Reverse proxy caches, 386–387
  - Reviewing
    - 360-degree reviews of leaders, 69
    - change effectiveness review, 177–178
    - daily incident review, 141–143
    - employee reviews of leadership, 69
    - incidents. *See* Incident management, incident review; Postmortem review.
    - problems, 142. *See also* Postmortem review.
    - quarterly incident review, 143, 146, 147
  - Rigor, business processes, 126–128
  - Risk factors, TAD, 302–303
  - Risk management
    - acute risk, 252–253
    - in change management, 173, 175
    - cumulative amount of changes, 253–254
    - human factor, 254
    - importance to scalability, 244–245
    - overall risk, 253–254
  - Risk management, measuring risk
    - ability to detect failure, 249
    - checklist of steps, 252
    - FMEA (Failure Mode and Effects Analysis), 249–251
    - gut feel method, 245–246
    - likelihood of failure, 249

- Risk management, measuring risk (*continued*)
    - severity of failure, 249
    - Total Risk Score, 249
    - traffic light method, 246–248
  - Risk profile, data center planning, 486–487
  - Roles and responsibilities. *See also specific roles.*
    - ambiguity, effects of, 22–23
    - assignment chart. *See* RASCI (Responsible, Approve, Support, Consult, Inform).
    - clarity, importance of, 22–23
    - defining, 23–24. *See also* RASCI (Responsible, Approve, Support, Consult, Inform).
    - delegation, 24
    - leadership, 19
    - management, 18–19
    - organizational example, 35–37
    - overlapping, 40
    - shareholder test, 24
    - voids, 41
  - Roles and responsibilities, executives. *See also* Corporate mindset.
    - balancing business and technical acumen, 28–29
    - business unit owners, 27
    - CEO (Chief Executive Officer), 25–26
    - CFO (Chief Financial Officer), 26–27
    - chief scalability officer. *See* CEO (Chief Executive Officer).
    - CIO (Chief Information Officer), 27–29
    - CTO (Chief Technology Officer), 27–29
    - general managers, 27
    - leading from the front, 28
    - P&L owners, 27
  - Roles and responsibilities, individual contributors
    - architects, 33
    - capacity planners, 35
    - crisis management, 157
    - database administrators, 34–35
    - incident managers, 34
    - infrastructure engineers, 34–35
    - network engineers, 34–35
    - operators, 34
    - QA analysts, 35
    - QA engineers, 35
    - scalability architects, 33
    - software engineers, 33–34
    - systems administrators, 34–35
  - Roles and responsibilities, organizational architecture, 29–30
    - capacity planning, 32
    - engineering, 30
    - infrastructure, 31
    - issue identification, 31
    - issue management, 31
    - production operations, 30–31
    - quality assurance, 31–32
  - Rollback
    - architectural principles, 201
    - barriers to, 281
    - cost considerations, 281–282
    - need for, 278–279
    - requirements checklist, 280
    - technology considerations, 281
    - version numbers, 281
    - window requirements, 279–280
  - Rule of three, architectural principles, 200–201
- ## S
- SaaS (Software as a Service), 427–428
  - Scalability
    - art *vs.* science, 2–3
    - need for, 3–4
  - Scalability architects, roles and responsibilities, 33
  - Scalability projects, determining headroom, 187–188
  - Scale Cube. *See* AKF Scale Cube.
  - Scale on demand, clouds, 431–432, 434
  - Scale out not up, architectural principles, 203, 207
  - Scaling cross functionally, JAD (Joint Architecture Design), 214–215
  - “Scared straight” method, 113–114
  - Scheduling, change management, 174–176
  - Schemas, separating data into, 362–365. *See also* AKF Scale Cube for databases, y-axis splits.
  - Schigel, Tim, 507–508
  - Science of scalability, 2–3
  - Scope, tradeoffs in business, 287

- Seasonality effect, determining headroom, 187
- Security, cloud computing drawback, 443
- Seeding teams, 98
- Seniority, as a factor in team membership, 47–48
- Separating data into schemas, 362–365. *See also* AKF Scale Cube for databases, y-axis splits.
- Servers, optimal number for data centers, 491–492
- Service behavior during failure, stress testing, 265
- Service providers, clouds, 435–436
- Services, splitting work responsibility by, 331–332
- Services affecting performance, identifying for stress testing, 265
- Session environments, saving, 403–404
- Session storage
  - avoiding, 403–404, 405
  - centralizing, 404, 405
  - decentralizing, 404, 405
- Set method, 382
- SETI (Search for Extraterrestrial Intelligence), 429
- SETI@home project, 429
- Sharding, definition, 311
- Shards, definition, 311
- Shared infrastructure, grid benefit, 457
- Shareholder test, 24
- Shareholder value, dilution by data cost, 415
- Shareholder values, leadership alignment with, 74–75
- ShareThis, case studies, 507–508
- Slavery, abolition of, 80
- Slivers, definition, 311
- SMART (Specific, Measurable, Attainable, Realistic, Timely)
  - criteria, postmortem review, 145
  - goals, 80–81, 83
  - guidelines, architectural principles, 198
- Software as a Service (SaaS), 427–428
- Software engineers, roles and responsibilities, 33–34
- Specific architectural principles, 198
- Speed
  - cloud computing benefit, 441–442
  - tradeoffs in business, 287
- Splitting databases by
  - customer, 365–367. *See also* Z-axis splits, databases.
  - customer geography, 365–367. *See also* Z-axis splits, databases.
  - requestor, 365–367. *See also* Z-axis splits, databases.
- Splitting teams. *See* Teams, splitting.
- Splitting work responsibility by. *See also* AKF Scale Cube.
  - application, 343–344. *See also* AKF Scale Cube for applications.
  - requestor or customer, 333–334, 344–347
  - resources, 331–332
  - services, 331–332
- Srivastava, Amitabh, 46
- Stability, definition, 165
- Staffing for scalability. *See also* Leadership; Management; Organizational design; Teams.
  - adding/removing people, 12, 13
  - importance of people, 10–11
- Stakeholders, organizing to match their interests, 13
- Standardization, business processes, 123
- Standards, organizational design influences on, 44–45. *See also* Business processes.
- State
  - within applications, AKF Scale Cube, 335
  - defining, 401
  - saving, 403–404
- Stateless systems, architectural principles, 202, 206–207
- Status communications, crisis management, 160–161
- Storage costs of data
  - backup storage, 413
  - initial cost of storage, 412–413
- Strategic advantage value of data, 415, 416–417
- Strategic competitive differentiation, building components *vs.* buying, 238
- Strategy-focused approach, building components *vs.* buying, 235–236, 237–240
- Strength, weaknesses, opportunities, threats (SWOT) analysis, 82

- Stress testing  
 analyzing data, 268  
 bottlenecks, identifying, 266  
 checklist of steps, 268–269  
 criticality of services, ranking, 265  
 data collection, 267  
 definition, 264  
 drawbacks, 269  
 environment, creating, 266  
 establishing a baseline, 265  
 executing tests, 267  
 failure, testing, 265  
 goals of, 264–265  
 interactivity of system services, 265  
 key services, identifying, 265  
 monitors, identifying, 267  
 negative testing, 264, 265  
 objectives, identifying, 264–265  
 positive testing, 264  
 recoverability, testing, 265  
 for scalability, 270–271  
 service behavior during failure, 265  
 services affecting performance,  
   identifying, 265  
 simulating a load, 267  
 system interactions, negative, 265
- Structured programming, 402
- Sun Grid Engine, 429
- Support persons, RASCI, 38
- Support services business, 109–110
- Swapping, excessive, 190
- Swim lanes  
 along natural barriers, 320–321  
 by the biggest sources of incidents, 320  
 by customer boundaries, 312–313  
 definition, 310, 311  
 isolating the money-maker, 320  
 by service boundaries, 313–314  
 splitting. *See* AKF Scale Cube.
- SWOT (strength, weaknesses, opportunities,  
 threats) analysis, 82
- Synchronization process, description, 393–394
- Synchronous calls. *See also* Asynchronous  
 design.  
*vs.* asynchronous, 395–401  
 example, 395
- Synchronous systems, scaling issues, 398–401
- Synchronous voice communications, crisis  
 management, 157–158
- Systems administrators, roles and  
 responsibilities, 34–35
- Systems monitoring, 477
- ## T
- TAA (technology agnostic architecture). *See*  
*also* Architectural principles; TAD  
 (technology agnostic design).  
 build *vs.* buy conflicts, 305  
 effects on availability, 306  
 implementing, rules for, 306–308  
 purpose of, 301
- TAD (technology agnostic design). *See also*  
 Architectural principles; TAA  
 (technology agnostic architecture).  
 build *vs.* buy decision, checklist, 304  
 build *vs.* buy decision, conflicts, 305  
 commoditization over time, 301–302  
 cost factors, 301–302  
 cultural aspects, 307–308  
 effects on availability, 306  
 implementation *vs.* architecture, 300  
 implementing, rules for, 306–308  
 overview, 300–301  
 purpose of, 301  
 risk factors, 302–303  
 scalability support, 303–305
- Tags, caches, 378
- Task management, 91–92
- Team focus *vs.* individual focus, leadership, 71
- Team growth *vs.* individual productivity,  
 13–15
- Team managers, role in crisis management,  
 155–156
- Team size, optimizing  
 check list, 54  
 communication breakdown, 50–51  
 communications, 48  
 effects on productivity, 50–51  
 growing or splitting, 52–54  
 importance of, 46–47  
 low boundary, 47  
 managerial experience levels, 47  
 managerial responsibilities, base level, 48

- micromanagement, 51
- morale issues, 50–51
- needs of the business, 49
- optimal range, 46–47
- overwork, 52
- sample analysis, 49–50
- seniority as a factor, 47–48
- too big, 49, 50–51. *See also* Teams, splitting.
- too small, 49, 50–51. *See also* Teams, growing.
- upper boundary, 47
- warning signs, 50–52
- Team structure
  - by management structure. *See* Team structure, functional organization; Team structure, matrix organization.
  - by purpose or function. *See* Team structure, functional organization.
  - silo approach. *See* Team structure, functional organization.
- Team structure, functional organization
  - benefits of, 56, 57
  - drawbacks, 56–57
  - organization chart, 55
- Team structure, matrix organization
  - cross-team communication, 59
  - drawbacks, 59
  - improving on the functional organization, 59–60
  - organization chart, 58
  - project ownership, 59
- Teams
  - adding/removing people, 12, 13
  - building, 93–94
  - causal roadmap to success, 84–86
  - empowerment, and scalability, 74
  - engineering, causal roadmap to success, 85
  - growing, 52–54
  - JAD (Joint Architecture Design), 218
  - managing. *See* Management, of teams.
  - operations, causal roadmap to success, 85–86
  - upgrading, 94–98
- Teams, splitting
  - check list, 54
  - failure domains, 53
  - naming a new manager, 52–53
  - relationship with business partners, 53
  - splitting the code base, 52
- Technology agnostic architecture (TAA). *See* TAA (technology agnostic architecture).
- Technology agnostic design (TAD). *See* TAD (technology agnostic design).
- Technology cycles, vicious *vs.* virtuous, 3
- Technology ride-alongs, 112
- Technology-as-product business, 109–110
- Telephone communications, crisis management, 157–158
- Terminating team members, 96
- Testable architectural principles, 198
- Testing, fault isolation, 321–322
- Third-party services
  - monitoring, calculating availability, 517–518
  - relying on, 360–362
  - software certification, cloud computing, 444–445
- 13th Amendment, U.S. Constitution, 80
- Thrashing, 190
- Three Magic Rules of Threes. *See* Data center planning, Three Magic Rules of Threes.
- 360-degree reviews of leaders, 69
- Tiered data storage solutions, 417–419
- Time to market
  - AKF Scale Cube for applications, 346
  - fault isolation, 315–316
  - x-axis splits, databases, 360–361
- Timeline considerations, AKF Scale Cube for databases, 373–374
- Timeline Phase, postmortem review, 145
- Timelines
  - incident review, 144
  - postmortem review, reviewing, 145
- Total Risk Score, 249
- Tradeoffs in business
  - cost, definition, 286
  - cost/speed/quality/scope triangle, 285–288
  - effects on scalability, 289–290
  - project triangle, 285–288
  - quality, definition, 286–287
  - scope, definition, 287
  - speed, definition, 287

Tradeoffs in business, documenting  
 ARB (Architecture Review Board), 228  
 JAD (Joint Architecture Design), 218  
 Tradeoffs in business, making decisions  
 checklist for, 294  
 comparing pros and cons, 291–292  
 decision matrix method, 292–295  
 gut feel method, 291  
 Traffic graphs, calculating availability,  
 518–519  
 Traffic-light risk management method,  
 246–248  
 Transaction growth. *See* X-axis splits; Y-axis  
 splits.  
 Transaction increases, scaling for, 341–342  
 Transaction processing time, database y-axis  
 splits, 364  
 Transforming data for storage, 419–420

## U

Undoing changes. *See* Rollback.  
 UNICORE (UNiform Interface to  
 Computing REsources), 429  
 Unmarshalling caches, 381  
 Unprofitable data, eliminating, 415  
 Updating caches. *See* Caches, refreshing.  
 U.S. Declaration of Independence, 76–77  
 U.S. Pledge of Allegiance, 76  
 User-experience monitoring, 476–477  
 Utilization of unused capacity, grid  
 benefit, 457

## V

Validation, change management, 176–177  
 Venn diagram, architectural principles, 196  
 Version numbers, rollback, 281  
 Vicious technology cycles, 3, 503  
 Virtualization, clouds  
 private, 434  
 public, 433  
 software for, 435–436  
 Virtualization, VMM (virtual machine  
 monitor), 433  
 Virtuous technology cycles, 3, 503  
 Vision. *See* Leadership, vision.

Vista. *See* Microsoft Vista.  
 VMM (virtual machine monitor), 433  
 Voids in roles and responsibilities, 41

## W

War room, crisis management, 158–159  
 Warning signs, sub-optimal team size, 50–52  
 Waterfall development, barrier conditions,  
 277–278  
 Webb, Maynard, 505–506  
 Weeding teams, 98  
 Whitman, Meg, 505–506  
 Window requirements, rollback, 279–280  
 Work flow, organizing for efficiency, 15–17  
 Work growth, by system or platform. *See*  
 X-axis splits; Y-axis splits.  
 Write-back method, 381  
 Write-through policy, 381  
 Writing to caches  
 dirty data, 381  
 set method, 382  
 write-back method, 381  
 write-through policy, 381

## X

XaaS (Everything as a Service), 427–428  
 x-axis, AKF Scale Cube, 328–330, 334–335  
 x-axis splits, applications  
 cost, 341–342  
 description, 341–342  
 fault isolation, 343  
 observing results, 353  
 scaling applications, cost, 341–342  
 uses for, 354  
 x-axis splits, databases. *See also* AKF Scale  
 Cube for databases.  
 capacity planning, 361  
 configuration management, 361  
 cost, 360, 361–362  
 data consistency, 360–362  
 data currency, 362  
 description, 358–359  
 increasing data size or amount, 361  
 pros and cons, 360–362  
 reliance on third parties, 360–362

replication delays, 359–360, 361  
summary of, 367–370  
time to market, 360–361  
*vs.* y-axis splits, 363

## Y

y-axis, AKF Scale Cube, 331–332, 335  
y-axis splits, applications. *See also* AKF Scale  
Cube for applications.  
cost, 344  
description, 343–344  
fault isolation, 345  
observing results, 353  
scaling applications, cost, 344  
uses for, 354  
y-axis splits, databases. *See also* AKF Scale  
Cube for databases.  
cost, 363, 364  
description, 362–365  
handling size and complexity, 364  
noun perspective, 363  
pros and cons, 363–364

purpose of, 362–363  
summary of, 367–370  
transaction processing time, 364  
*vs.* x-axis splits, 363

## Z

z-axis, AKF Scale Cube, 333–334, 335  
z-axis splits, applications. *See also* AKF Scale  
Cube for applications.  
cost, 345–347  
description, 344–347  
fault isolation, 346  
observing results, 353  
scaling applications, cost, 345–347,  
347–348  
uses for, 354  
z-axis splits, databases. *See also* AKF Scale  
Cube for databases.  
cost, 366, 367  
description, 365–367  
pros and cons, 366  
summary of, 367–370