THE **LEAN** SERIES ERIC RIES, SERIES EDITOR

Jez Humble, Joanne Molesky & Barry O'Reilly

LEAN ENTERPRISE

How High Performance Organizations Innovate at Scale

O'REILLY®

Praise for *Lean Enterprise*

"This book is *Reengineering the Corporation* for the digital age. It is destined to be the classic, authoritative reference for how organizations plan, organize, implement, and measure their work. *Lean Enterprise* describes how organizations can win in the marketplace while harnessing and developing the capabilities of employees. Any business leader who cares about creating competitive advantage through technology and building a culture of innovation needs to read this book."

— Gene Kim, co-author of *The Phoenix Project: A Novel About IT, DevOps, and Helping Your Business Win*, founder and former CTO of Tripwire, Inc.

"This book is a godsend for anyone who's tried to change their organization and heard: 'It's OK for the little guy, but we're too big/regulated/complex to work like that here.' Humble, Molesky, and O'Reilly have written an easy-to-read guide that demystifies the success of Lean organizations in a way that everyone can understand and apply. *Lean Enterprise* provides a pragmatic toolkit of strategies and practices for establishing high performing organizations. It should be required reading for every executive who understands that we're all in the technology business now."

— Stephen Foreshew-Cain, COO, UK Government Digital Service

"To thrive in the digital world, transformation must be more than technology driven—everyone within the organization must collectively work together to adapt. This book provides an essential guide for all leaders to change the way they deliver value to customers."

— Matt Pancino, CEO, Suncorp Business Services

"This is the book I've been waiting for—one that takes on the hardest questions in bringing Lean approaches to the enterprise. The authors provide solutions that are valuable even in low trust environments."

— Mark A. Schwartz (*@schwartz_cio*)

"This book integrates into a compelling narrative the best current thinking about how to create great software-intensive products and services. The approach in this book is both challenging and disciplined, and some organizations will be unable to imagine following this path. But those who make the journey will find it impossible to imagine ever going back—and if they happen to be a competitor, they are well positioned to steal both your market and your people. Ignore this book at your own risk."

— Mary Poppendieck, co-author of *The Lean Mindset* and the Lean Software Development series

"My job is to support people in practicing a scientific pattern, to help reshape thinking and working habits in business, politics, education, and daily life. The 21st century is increasingly demanding a way of working that's cognitively complex, interpersonal, iterative, and even entrepreneurial. With *Lean Enterprise*, Jez Humble, Joanne Molesky, and Barry O'Reilly explain how software can and is leading the way to transforming our ways of working, which can change our ways of thinking and help us adapt to the emerging world around us."

— Mike Rother, author of *Toyota Kata*

"Nearly all industries and institutions are being disrupted through the rapid advance of technology, guided by the inspired vision of individuals and teams. This book clearly explains how the disciplines of Lean, Agile, Kata, Lean Startup, and Design Thinking are converging through the unifying principles of an adaptive learning organization."

— Steve Bell, Lean Enterprise Institute faculty, author of *Lean IT* and *Run Grow Transform*

"Building software the right way is a challenging task in and of itself, but *Lean Enterprise* goes beyond the technology considerations to guide organizations on how to quickly build the right software to deliver expected business results in a low risk fashion. This is a must read for any organization that provides software based services to its customers."

— Gary Gruver, VP of Release, QE, and Operations for Macys.com

"To compete in the future businesses need to be skilled at understanding their customers and taking the validated learnings to market as quickly as possible. This requires a new kind of adaptive and learning organization—the lean enterprise. The journey starts here in this book!"

— John Crosby, Chief Product and Technology Officer, lastminute.com

"Rapid advancements in technology are creating unparalleled rates of disruption. The rules of the disruption game have changed, and many organizations wonder how to compete as new giants emerge with a different approach to serving their customers. This book provides an essential guide to those that have come to the realization that they have to change to regain an innovative competitive advantage but are unsure where to start."

— Jora Gill, Chief Digital Officer, The Economist

``*Lean Enterprise* was the book I gave my leadership team to get everyone on the same page about how we can challenge the status quo, remove roadblocks, and out-innovate our competition. By leveraging the continual insights we get from co-creating with customers, our people, and data, we now have so many additional new ways to grow our business.''

— Don Meij, CEO, Domino's Pizza Enterprises Ltd.

"While agile and lean methods have had a big impact on software delivery, their true potential only comes as they have a broader impact on enterprises of all sizes. In this book, Jez, Joanne, and Barry have set out what those changes look like—a realistic vision of how future companies will make today's look like cassette tape players."

— Martin Fowler, Chief Scientist, ThoughtWorks

"This is an important book. It takes an informed and informative look at the fundamentals that need to shift to start building organizations capable of continuous learning and improvement. It moves well beyond the technical to the organizational. *Lean Enterprise* is a must-read for existing and emerging leaders seeking to ensure their company's ongoing success."

— Jeff Gothelf, author of *Lean UX*, and Principal of Neo Innovation

"I was telling everyone to get this book for a year before it was finished. It documents the path being taken by the leading lean enterprises and the fat ones will be wiped out by the lean ones in the years to come."

— Adrian Cockcroft (@*adrianco*)

Lean Enterprise

Jez Humble, Joanne Molesky, and Barry O'Reilly

Beijing • Cambridge • Farnham • Köln • Sebastopol • Tokyo O'REILLY®

Lean Enterprise
by Jez Humble, Joanne Molesky, and Barry O'Reilly

Printed in the United States of America.

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

O'Reilly books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (*http://safaribooksonline.com*). For more information, contact our corporate/institutional sales department: 800-998-9938 or *corporate@oreilly.com*.

Editors: Mary Treseler and Angela Rufino
Production Editor: Kara Ebrahim
Copyeditor: Dmitry Kirsanov
Proofreader: Alina Kirsanova

Indexers: Dmitry Kirsanov and Alina Kirsanova
Interior Designer: David Futato
Cover Designer: Ellie Volckhausen
Illustrators: Rebecca Demarest and Peter Staples

Revision History for the First Edition
2014-12-01: First Release

See *http://oreilly.com/catalog/errata.csp?isbn=9781449368425* for release details.

The O'Reilly logo is a registered trademark of O'Reilly Media, Inc. *Lean Enterprise*, the cover image, and related trade dress are trademarks of O'Reilly Media, Inc.

978-1-449-36842-5

[CW]

This book is dedicated to all of you who have (to paraphrase Admiral Grace Hopper) asked for forgiveness, not permission, in the pursuit of perfection, and to all the leaders committed to creating organizations where everybody knows what the right thing is, and you don't need anyone's permission to do it.

Contents

PART III: EXPLOIT

PART IV: TRANSFORM

Preface

> Software is eating the world.
>
> — Marc Andreesen

> In an industrial company, avoid software at your own peril . . . a software company could disintermediate GE someday, and we're better off being paranoid about that.
>
> — Jeff Immelt

> You are a fool if you do just as I say. You are a greater fool if you don't do as I say. You should think for yourself and come up with better ideas than mine.
>
> — Taiichi Ohno, Workplace Management

In this book we show how to grow organizations which can innovate rapidly in response to changing market conditions, customer needs, and emerging technologies.

Companies live and die on their ability to discover new businesses and create ongoing value for customers. This has always been true, but never more so than in the past few years. Competitive pressure is increasing, fueled by rapid changes in technology and society. As Deloitte's Shift Index shows, the average life expectancy of a Fortune 500 company has declined from around 75 years half a century ago to less than 15 years today. Professor Richard Foster of Yale University estimates that "by 2020, more than three-quarters of the S&P 500 will be companies that we have not heard of yet."[1] The long-term survival of

1 *http://www.bbc.co.uk/news/business-16611040*

any enterprise depends on its ability to understand and harness the cultural and technical forces that continue to accelerate innovation cycles.

First, the Internet and social media have provided consumers with powerful tools to inform the decisions they make. These tools also give smart organizations new ways to discover and engage with users and customers. Enterprises that use design thinking and user experience (UX) design strategically to delight customers at each step of their interaction with the organization have thrived: research shows companies which apply UX design in this way experience faster growth and higher revenues.[2]

Second, advances in technology and process have made it possible to build, evolve, and scale disruptive products and services rapidly and with little capital investment. Small teams across the world prototype new software-based products in days or weeks, using free or cheap services and infrastructure, and then rapidly evolve those that gain traction. In the near future, the ubiquity of cheap, powerful networked embedded devices will enable us to prototype and evolve a wider variety of products cheaply on similarly short cycles. As 3D printing becomes cheaper and faster and begins to handle a wider variety of materials, we will create and deliver an enormous variety of customized products on demand.

Software has three characteristics which enable this kind of rapid innovation. First, it's relatively inexpensive to prototype and evolve ideas in software. Second, we can actually use such prototypes from an early stage in their evolution. Finally, in the course of creating these prototypes, we can discover a great deal about what customers find valuable and incorporate it back into our design—accelerating the rate at which we can test new ideas with users, collect feedback, and use it to improve our products and businesses.

Meanwhile, the relentless march of miniaturization (embodied in Moore's Law)[3] has enabled incredibly powerful computers to become tiny and find their way into everything, with software at center stage. In a Forbes article titled "Now Every Company Is A Software Company," David Zanca, senior vice president for information technology at FedEx, describes himself as running "a software company inside of FedEx." Venkatesh Prasad, senior technical leader at Ford, describes his company as a maker of "sophisticated computers-on-wheels." Ben Wood of CCS Insight notes that Nokia "went through this incredible decade of innovation in hardware, but what Apple saw was that all you needed was a rectangle with a screen, and the rest was all about the

2 *Evaluation of the Importance of Design*, Danish Design Center, 2006.

3 In 1965 Gordon Moore, co-founder of Intel, predicted that the density of integrated circuits would double approximately every two years.

software."[4] As a result of this shift in thinking about software, companies, including IT outsourcing pioneers GE and GM, are taking software development back in-house. As we discuss in Chapter 15, the UK government has followed suit. As reported by *The Economist*:[5]

> GM's reasons for doing this may well apply to many other firms too. "IT has become more pervasive in our business and we now consider it a big source of competitive advantage," says Randy Mott, GM's Chief Information Officer, who has been responsible for the reversal of the outsourcing strategy. While the work was being done by outsiders, he said most of the resources that GM was devoting to IT were spent on keeping things going as they were rather than on thinking up new ways of doing them. The company reckons that having its IT work done mostly in-house and nearby will give it more flexibility and speed and encourage more innovation.

The business world is moving from treating IT as a utility that improves internal operations to using rapid software- and technology-powered innovation cycles as a competitive advantage. This has far-reaching consequences. The traditional program and project management models we have used for IT are unsuited to rapid innovation cycles. However, they are deeply embedded in the way we manage everything from operations and customer service to budgeting, governance, and strategy. The elements of a suitable product-centric paradigm that works at scale have all emerged in the last 10 years, but they have not yet been connected and presented in a systematic way. This book aims to fill this gap, providing inspiration from organizations that have successfully adopted these ideas. More importantly, we have made a detailed inquiry into the culture of high performance, which is the critical factor enabling rapid innovation at scale.

Why Did We Write This Book?

All of the authors are experienced working in both enterprises and startups, and we have set out to present a pragmatic and systematic approach to innovation and transformation that works effectively in an enterprise context. We have addressed not just how high-performing organizations develop products, but how companies that are working towards higher performance can adopt these techniques in an incremental, iterative, low-risk way.

---

4 *http://www.bbc.co.uk/news/technology-23947212*; in our opinion, this is the key insight behind Microsoft's acquisition of Nokia.

5 *The Economist Special Report: Outsourcing and Offshoring*, 406, no. 8819, 19 January 2013.

We wrote the book because of our frustration at the state of the industry. The techniques and practices we describe are not new, and they are known to work. However, they are not yet mainstream, and are often implemented piecemeal, leading to local, rather than systemic, improvements. As a result, companies toil at building—at huge cost—products, services, and businesses that do not deliver the expected value to customers.

When *Continuous Delivery* (Addison-Wesley) and *The Lean Startup* (Crown Business) were published, we saw an enormous amount of demand from people working in enterprises who wanted to adopt the practices described in these books. A large number of companies have achieved measurable benefit from using the practices we discuss, resulting in delivery of higher-quality products to market faster, increased customer satisfaction, and higher returns on investment. This comes with reduced cost and risk as well as happier employees who are no longer working unsustainable hours and have the opportunity to harness their creativity and passion at work.

However, everyone finds it difficult to implement these ideas successfully. In most cases it was impossible to realize anything more than incremental improvements because only part of the organization changed—and that part still needed to work with the rest of the organization, which expected them to behave in the traditional way. Thus we describe how successful companies have rethought everything from financial management and governance, to risk and compliance, to systems architecture, to program, portfolio, and requirements management in the pursuit of radically improved performance.

This book presents a set of patterns and principles designed to help you implement these ideas. We believe that every organization is different and will have different needs, so we don't provide rules on how to implement particular practices. Instead, we describe a heuristic approach to implementation that emphasizes the importance of experimentation in order to learn how your organization can best adopt these ideas and improve. This approach takes longer, but it has the advantages of showing measurable benefits faster and reducing the risk of change. It also enables your organization and people to learn for themselves what works best.

We hope you will find value in this book. The most dangerous attitude would be: "These are good ideas, *but they cannot work in our organization.*" As Taiichi Ohno, the father of the Toyota Production System, said:[6]

> Whether top management, middle management, or the workers who actually do the work, we are all human, so we're like walking

6 [ohno12]

> misconceptions, believing that the way we do things now is the best way. Or perhaps you do not think it is the best way, but you are working within the common sense that "We can't help it, this is how things are."

You will face obstacles adopting the ideas in this book. When you read the case studies, you will likely see reasons why the described approach may not work in your organization. Do not turn obstacles into objections. Treat what you read here as an inspiration for your own efforts, not as recipes to be followed without deviation. Look for obstacles constantly and treat them as opportunities to experiment and learn. To quote Ohno again:[7]

> Kaizen [improvement] opportunities are infinite. Don't think you have made things better than before and be at ease…This would be like the student who becomes proud because they bested their master two times out of three in fencing. Once you pick up the sprouts of kaizen ideas, it is important to have the attitude in our daily work that just underneath one kaizen idea is yet another one.

Opportunities to improve lie everywhere—not just in the products or services we build but in the way we behave and interact and, most importantly, in the way we think.

Who Should Read This Book?

We wrote this book primarily for leaders and managers. The book focuses on principles and patterns that can be applied in any domain in any type of organization.

Our intended audience includes:

- Executives interested in strategy, leadership, organization culture, and good governance
- Directors of IT, both for applications and for infrastructure and operations
- Anyone working in program or project management, including members of the PMO
- People in finance and accounting or in governance, regulation, and compliance who are involved in delivery

7 [ohno12]

- CMOs, product managers, and others involved in designing products and services that involve software development

Anyone working on delivery teams should also find this book valuable—but don't expect any deep discussion of engineering practices, such as how to write maintainable functional acceptance tests, automate deployment, or manage configuration. Those topics are discussed in much more depth in *Continuous Delivery*.

This book is particularly targeted at people working in medium and large organizations who realize they must think differently about strategy, culture, governance, and the way they manage products and services in order to succeed. That's not to say that smaller organizations won't find the book useful—just that some of the material may not be applicable to them at this stage in their evolution.

One of our goals was to keep the book relatively short, concise, and practical. In order to do that, we decided not to spend a lot of time discussing the theoretical models that drive the principles and practices we describe. Instead, we have presented some foundational principles from these fields so you can understand the basic theoretical underpinnings; then we describe the practical applications of these theories. We also provide references to further reading for those who are interested.

We are also careful not to offer detailed guidance on which software tools to use and how to use them. This is for two reasons. First, we think that tool choice is actually not a tremendously important decision (so long as you avoid the bad ones). Many organizations moving to agile methodologies spend an undue amount of time on tool choice hoping to magically solve their underlying problems. But the most common failure mode for such organizations is their inability to change their organizational culture, not the availability of good tools. Secondly, information on particular tools and processes quickly goes out of date. There are plenty of good tools (including many open source ones) and literature on how to use them. In this book we focus on strategies to help your organization succeed, regardless of the tools you choose.

Conspectus

Part I of the book introduces the main themes of the book: culture, strategy, and the lifecycle of innovations. In Part II we discuss how to explore new ideas to gather data so you can quickly evaluate which ones will provide value or see a sufficiently rapid uptake. Part III covers how to exploit validated ideas—those that emerge from the crucible of exploration—at scale, and also presents a systematic approach to improving the way we run large programs of work. Finally, Part IV shows how enterprises can grow an environment that fosters

learning and experimentation, with a focus on culture, governance, financial management, IT, and strategy.

Everybody should read Part I. Readers should then feel free to dip into the chapters that interest them. However it's worth reading Chapter 3, Chapter 6, and Chapter 7 before proceeding to Part IV since it builds on concepts presented in those chapters.

Safari® Books Online

NOTE

Safari Books Online is an on-demand digital library that delivers expert content in both book and video form from the world's leading authors in technology and business.

Technology professionals, software developers, web designers, and business and creative professionals use Safari Books Online as their primary resource for research, problem solving, learning, and certification training.

Safari Books Online offers a range of product mixes and pricing programs for organizations, government agencies, and individuals. Subscribers have access to thousands of books, training videos, and prepublication manuscripts in one fully searchable database from publishers like O'Reilly Media, Prentice Hall Professional, Addison-Wesley Professional, Microsoft Press, Sams, Que, Peachpit Press, Focal Press, Cisco Press, John Wiley & Sons, Syngress, Morgan Kaufmann, IBM Redbooks, Packt, Adobe Press, FT Press, Apress, Manning, New Riders, McGraw-Hill, Jones & Bartlett, Course Technology, and dozens more. For more information about Safari Books Online, please visit us online.

How to Contact Us

Please address comments and questions concerning this book to the publisher:

O'Reilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472
800-998-9938 (in the United States or Canada)
707-829-0515 (international or local)
707-829-0104 (fax)

We have a web page for this book, where we list errata, examples, and any additional information: *http://bit.ly/lean-enterprise-book*.

To comment or ask technical questions about this book, send email to *bookquestions@oreilly.com*.

For more information about our books, courses, conferences, and news, see our website at *http://www.oreilly.com*.

Find us on Facebook: *http://facebook.com/oreilly*

Follow us on Twitter: *http://twitter.com/oreillymedia*

Watch us on YouTube: *http://www.youtube.com/oreillymedia*

Acknowledgments

Many people have contributed to this book. In particular, we are deeply grateful to the following people who provided detailed reviews of early drafts or individual chapters (alphabetically by first name): Adrian Cockcroft, Amy McLeod, Andy Pittaway, Bas Vodde, Ben Williams, Bjarte Bogsnes, Brett Ansley, Carmen Cook, Charles Betz, Chris Cheshire, Courtney Hemphill, Dan North, Darius Kumana, David Tuck, Don Reinertsen, Gary Gruver, Gene Kim, Ian Carroll, James Cook, Jean-Marc Domaingue, Jeff Gothelf, Jeff Patton, Jim Highsmith, Joe Zenevitch, John Allspaw, John Crosby, Jonathan Thoms, Josh Seiden, Kevin Behr, Kief Morris, Kraig Parkinson, Lane Halley, Lee Nicholls, Lindsay Ratcliffe, Luke Barrett, Marc Hofer, Marcin Floryan, Martin Fowler, Matt Pancino, Michael Orzen, Mike Rother, Pat Kua, Randy Shoup, Ranjan Sakalley, Salim Virani, Steve Bell, Tom Barker, Tristan Kromer, and Will Edelmuth. Thank you so much. The ideas we present came from a wide variety of sources, and were winnowed and refined through innumerable workshops, talks, and discussions with people working in an enormous variety of organizations across the world. Thanks to all of you who participated in those discussions and gave us the benefit of your experiences and feedback. We'd like to extend special thanks to our fabulous editorial and production team at O'Reilly: Mary Treseler, Angela Rufino, Allyson MacDonald, Kara Ebrahim, and Dan Fauxsmith. Special thanks are also due to Peter Staples for creating almost all of the gorgeous diagrams in the book. Steve Bell, John Kordyback, Scott Buckley, and Gareth Rushgrove provided case studies for this book: thanks so much for your contributions and insight. Finally, Dmitry Kirsanov and Alina Kirsanova did characteristically thorough, detailed, and high-quality work copyediting, proofreading, and indexing the book—thank you.

Jez started working on this book as an excuse to stay home after his second daughter, Reshmi, was born. Reshmi and her sister, Amrita, have taught him the joy of disruption throughout by playing pranks and co-creating many new adventures that provoked both new insights and helpless laughter. Rani, his beautiful, brilliant wife, kept it real throughout even when it felt relentless, for which she has his undying gratitude, love, and admiration. He thanks his mum for her encouragement and support, particularly when he had to write during visits. Jez would like to thank his co-authors Joanne and Barry for moderating his command-and-control tendencies and making this book a truly

collaborative exercise. It would have been a very different—and much poorer—book without you. He would like to thank his colleagues at Chef for providing inspiration and support, and for living the dream of stirring up delight in the pursuit of a world-class product and customer experience. He also wants to thank his previous employer, ThoughtWorks, for providing a unique, mindful home for innovators and tinkerers, many of whose ideas populate these pages. Finally, special thanks to Chris Murphy, Chad Wathington, David Rice, Cyndi Mitchell, Barry Crist, and Adam Jacob for their support of this book.

Joanne really didn't understand what she had agreed to when Jez Humble and Martin Fowler convinced her to collaborate on a book about the next steps for Continuous Delivery. As time progressed (over two and a half years) and the book evolved into what it is today, there are a lot of people who provided support, encouragement, and complete trust in her capabilities to finish this work. John, Joanne's husband, lifetime partner, and best friend, provided encouragement and unending understanding during those guilt-ridden weekends and evenings when "the book" distracted her from fun activities. Her colleagues and the leadership team at ThoughtWorks provided all that she needed to research and write this work, in particular David Whalley, Chris Murphy, and the ThoughtWorks Australia leadership team who hired her—because they understood how important it is for something as command-and-control as security, risk, and compliance to fit with agile and lean delivery practices. Last, but not least, she would like to acknowledge her co-authors and good friends Barry and Jez, who taught her about perseverance, collaboration, and true trust in each other.

Barry could not have written this book without Qiu Yi, his life editor, partner, and wife. Her passion, persistence, and patience smooths his edges. Her compassion knows no end. His parents, Niall and Joan, have always believed in him, providing support and making personal sacrifices to enable him to reach for his goals. He could not ask for better role models; their principles and values have shaped his own, and for that he is grateful. He misses his brothers and sisters. The time they spend together is always precious and too short. His entire family is close to his heart and never far from his thoughts. He has been inspired by many friends, colleagues, and storytellers in his life and career; their conversations, lessons, and knowledge is captured here. Thank you for exposing him to it. When he wrote his first blog and pressed publish, he never imagined the outcome would lead him here. The encouragement, collaboration, and calibration of Jez and Joanne have taught him much more than how to craft ideas into words—he's grown with their guidance.

PART I

ORIENT

> The purpose of an organization is to enable ordinary human beings to do extraordinary things.
>
> — Peter Drucker

> Shareholder value is the dumbest idea in the world…[it is] a result, not a strategy…Your main constituencies are your employees, your customers, and your products.[1]
>
> — Jack Welch

We begin by offering our definition of an enterprise: "a complex, adaptive system composed of people who share a common purpose." We thus include nonprofits and public sector companies as well as corporations. We will go into more detail on complex, adaptive systems in Chapter 1. However, the idea of a common purpose known to all employees is essential to the success of an enterprise. A company's purpose is different from its vision statement (which describes what an organization aspires to become) and its mission (which describes the business the organization is in). Graham Kenny, managing director of consultancy *Strategic Factors*, describes the purpose of an organization as what it does *for someone else*, "putting managers and employees *in*

1 *http://on.ft.com/1zmWBMd*

customers' shoes."[2] He cites as examples the Kellogg food company ("Nourishing families so they can flourish and thrive") and the insurance company IAG ("To help people manage risk and recover from the hardship of unexpected loss"), to which we add our favorite example: SpaceX, "founded in 2002 by Elon Musk to revolutionize space transportation and ultimately make it possible for people to live on other planets."[3]

Creating, updating, and communicating the company's purpose is the responsibility of the enterprise's executives. Their other responsibilities include creating a strategy through which the company will achieve its purpose and growing the culture necessary for that strategy to succeed. Both strategy and culture will evolve in response to changes in the environment, and leaders are responsible for directing this evolution and for ensuring that culture and strategy support each other to achieve the purpose. If leaders do a good job, the organization will be able to adapt, to discover and meet the changing customer needs, and to remain resilient to unexpected events. This is the essence of good governance.

In the context of corporations, the idea of a common purpose other than profit maximization may seem quaint. For many years, the conventional wisdom held that corporate executives should focus on maximizing shareholder value, and this goal was reinforced by compensating executives with stocks.[4] However, these strategies have a number of flaws. They create a bias towards short-term results (such as quarterly earnings) at the expense of longer-term priorities such as developing the capabilities of employees and the relationships with customers. They also tend to stifle innovation by focusing on tactical actions to reduce costs in the short term at the expense of riskier strategies that have the potential to provide a higher payoff over the lifetime of the organization, such as research and development or creating disruptive new products and services. Finally, they often ignore the value of intangibles, such as the capabilities of employees and intellectual property, and externalities such as the impact on the environment.

Research has shown that focusing only on maximizing profits has the paradoxical effect of *reducing* the rate of return on investment.[5] Rather, organizations

---

2 *http://bit.ly/1zmWArB*

3 In the copious free time left over from SpaceX, Musk co-founded Tesla Motors along with "a group of intrepid Silicon Valley engineers who set out to prove that electric vehicles could be awesome."

4 This strategy originates from Jensen and Meckling's "Theory of the Firm" (*Journal of Financial Economics*, 3, no. 4, 1976).

5 John Kay's *Obliquity* (Penguin Books) provides detailed research and analysis supporting what he describes as the "profit-seeking paradox."

succeed in the long term through developing their capacity to innovate and adopting the strategy articulated by Jack Welch in the above epigraph: focusing on employees, customers, and products. Part I of this book sets out how to achieve this.

CHAPTER 1

Introduction

> It's possible for good people, in perversely designed systems, to casually perpetrate acts of great harm on strangers, sometimes without ever realizing it.
>
> — Ben Goldacre

On April 1, 2010, California's only motor vehicle plant, New United Motor Manufacturing, Inc. (NUMMI), shut down. NUMMI, which opened in 1984, had been a joint venture between GM and Toyota. Both companies stood to benefit from the partnership. Toyota wanted to open a plant in the US to escape import restrictions threatened by the US Congress in reaction to the inexorably falling market share of US auto manufacturers. For GM, it was a chance to learn how to build small cars profitably and to study the Toyota Production System (TPS) that had enabled Japanese auto manufacturers to consistently deliver the highest quality in the industry at costs that undercut those of US manufacturers.[1]

For the joint venture, GM chose the site of their shuttered Fremont Assembly plant. GM's Fremont plant was one of their worst in terms of both the quality of the cars produced and the relationship between managers and workers. By the time the plant closed in 1982, labor relations had almost completely broken down, with workers drinking and gambling on the job. Incredibly, Toyota agreed to the demand of United Auto Workers' negotiator Bruce Lee to rehire

1 The story of the NUMMI plant is covered comprehensively in *This American Life*, episode 403: *http://www.thisamericanlife.org/radio-archives/episode/403/*, from which all the direct quotes are taken.

the union leaders from Fremont Assembly to lead the workforce at NUMMI. The workers were sent to Toyota City in Japan to learn the TPS. Within three months, the NUMMI plant was producing near-perfect quality cars—some of the best quality in America, as good as those coming from Japan—at much lower cost than Fremont Assembly had achieved. Lee had been right in his bet that "it was the system that made it bad, not the people."

Much has been written about the TPS, but one recurring theme, when you listen to the Fremont Assembly workers who ended up at NUMMI, is teamwork. It might seem banal, but it was an incredibly powerful experience for many of the UAW employees. The TPS makes building quality into products the highest priority, so a problem must be fixed as soon as possible after it's discovered, and the system must then be improved to try and prevent that from happening again. Workers and managers cooperate to make this possible. The moment a worker discovers a problem, he or she can summon the manager by pulling on a cord (the famous *andon* cord). The manager will then come and help to try and resolve the problem. If the problem cannot be resolved within the time available, the worker can stop the production line until the problem is fixed. The team will later experiment with, and implement, ideas to prevent the problem from occurring again.

These ideas—that the primary task of managers is to help workers, that workers should have the power to stop the line, and that they should be involved in deciding how to improve the system—were revolutionary to the UAW employees. John Shook, the first American to work in Toyota City, who had the job of training the NUMMI workers, reflects that "they had had such a powerful emotional experience of learning a new way of working, a way that people could actually work together collaboratively—as a team."

The way the TPS works is in sharp contrast to the traditional US and European management practice based on the principles of Frederick Winslow Taylor, the creator of *scientific management*. According to Taylor, the job of management is to analyze the work and break it down into discrete tasks. These tasks are then performed by specialized workers who need understand nothing more than how to do their particular specialized task as efficiently as possible. Taylorism fundamentally thinks of organizations as machines which are to be analyzed and understood by breaking them down into component parts.

In contrast, the heart of the TPS is creating a high-trust culture in which everybody is aligned in their goal of building a high-quality product on demand and where workers and managers collaborate across functions to constantly improve—and sometimes radically redesign—the system. These ideas from the TPS—a high-trust culture focused on continuous improvement (*kaizen*), powered by alignment and autonomy at all levels—are essential to building a large organization that can adapt rapidly to changing conditions.

A key part of the success of the TPS is in its effect on workers. Taylorism makes workers into cogs in a machine, paid simply to perform preplanned actions as quickly as possible. The TPS, instead, requires workers to pursue mastery through continuous improvement, imbues them with a higher purpose —the pursuit of ever-higher levels of quality, value, and customer service—and provides a level of autonomy by empowering them to experiment with improvement ideas and to implement those that are successful.

Decades of research have shown that these *intrinsic motivators* produce the highest performance in tasks which require creativity and trial-and-error—where the desired outcome cannot be achieved simply by following a rule.[2] In fact, extrinsic motivators such as bonuses and rating people in performance reviews actually *decrease* performance in such nonroutine work.[3] Rick Madrid, who worked at the Fremont plant both before and during the NUMMI era, says of the TPS that "it changed my life from being depressed, bored—and like my son said, it changed my attitude. It changed me all for the better." Giving people pride in their work rather than trying to motivate them with carrots and sticks is an essential element of a high-performance culture.[4]

Although the principles at the heart of the TPS might seem relatively straightforward, they were very hard to adopt. Indeed, GM utterly failed in taking what it had achieved at NUMMI and reproducing it in other GM plants. Some of the biggest obstacles were changes to the organizational hierarchy. The TPS does away with the concept of seniority in which union workers are assigned jobs based on how many years of service they have, with the best jobs going to the most senior. Under the TPS, everybody has to learn all the jobs required of their team and rotate through them. The TPS also removes the visible trappings and privileges of management. Nobody wore a tie at the NUMMI plant —not even contractors—to emphasize the fact that everybody was part of the same team. Managers did not receive perks accorded to them at other GM plants, such as a separate cafeteria and car park.

Finally, attempts to improve quality ran up against organizational boundaries. In the TPS, suppliers, engineers, and workers collaborate to continuously

---

2 Behavioral scientists often classify work into two types: routine tasks where there is a single correct result that can be achieved by following a rule are known as *algorithmic*, and those that require creativity and trial-and-error are called *heuristic*.

3 Decades of studies have repeatedly demonstrated these results. For an excellent summary, see [pink].

4 Indeed one of W. Edwards Deming's "Fourteen Points For The Transformation Of Management" is "Remove barriers that rob people in management and in engineering of their right to pride of workmanship. This means, *inter alia*, abolishment of the annual or merit rating and of management by objective" [deming], p. 24.

improve the quality of the parts and to make sure workers have the tools they need to do their job. This worked at NUMMI because the engineers were in-house and the parts came from Japanese suppliers that had a collaborative relationship with Toyota. In the US supply chain, things were different. If the parts that came in to GM assembly plants were of poor quality, or didn't fit, there was simply no mechanism to fix the problem.

Ernie Schaefer, manager of GM's Van Nuys plant—which faced many of the same problems as Fremont Assembly—describes what was different about NUMMI: "You can see a lot of things different. But the one thing you don't see is the system that supports the NUMMI plant. I don't think, at that time, anybody understood the large nature of this system. General Motors was a kind of throw it over the wall organization. Each department, we were very compartmentalized, and you design that vehicle, and you'd throw it over the wall to the manufacturing guys." This is the legacy of a Taylorist management approach. The TPS exists—and can only succeed—within an ecosystem of organizational culture, supplier relations, financial management, HR, and governance designed around its philosophy.

GM tried to implement the TPS at Van Nuys, but failed. Workers and managers rebelled in the face of changes in status and behavior that were required of them, despite the threat of closure (which was ultimately carried out). According to Larry Spiegel, a veteran of NUMMI who had been sent to Van Nuys to help implement the TPS, people at the plant simply didn't believe the threats to shut it down: "There were too many people convinced that they didn't need to change."

This lack of urgency acted as a barrier to adoption across GM—and is perhaps the biggest obstacle to organizational change in general.[5] The US division of GM took about 15 years to decide they needed to seriously prioritize implementing the TPS, and a further 10 years to actually implement it. By this time any competitive advantage they could have gained was lost. GM went bankrupt and was bailed out by the US government in 2009, at which point it pulled out of NUMMI. Toyota shut down the NUMMI plant in 2010.

The story of NUMMI is important because it illustrates the main concern of this book—growing a lean enterprise, such as Toyota—and many of the common obstacles. Toyota has always been very open about what it is doing, giving public tours of its plants, even to competitors—partly because it knows that what makes the TPS work is not so much any particular practices but the

5 John Kotter, author of *Leading Change*, says, "a majority of employees, perhaps 75 percent of management overall and virtually all of the top executives, need to believe that considerable change is absolutely essential" [kotter], p. 51.

culture. Many people focus on the practices and tools popularized by the TPS, such as the *andon* cords. One GM vice president even ordered one of his managers to take pictures of every inch of the NUMMI plant so they could copy it precisely. The result was a factory with *andon* cords but with nobody pulling them because managers (following the principle of extrinsic motivation) were incentivized by the rate at which automobiles—of *any* quality—came off the line.

A Lean Enterprise Is Primarily a Human System

As the pace of social and technological change in the world accelerates, the lean approach pioneered by Toyota becomes ever more important because it sets out a proven strategy for thriving in uncertainty through embracing change. The key to understanding a lean enterprise is that it is primarily a *human* system. It is common for people to focus on specific practices and tools that lean and agile teams use, such as Kanban board, stand-up meetings, pair programming, and so forth. However, too often these are adopted as rituals or "best practices" but are not seen for what they really are—*countermeasures* that are effective within a particular context in the pursuit of a particular goal.

In an organization with a culture of continuous improvement, these countermeasures emerge naturally within teams and are then discarded when they are no longer valuable. The key to creating a lean enterprise is to enable those doing the work to solve their customers' problems in a way that is aligned with the strategy of the wider organization. To achieve this, we rely on people being able to make local decisions that are sound at a strategic level—which, in turn, relies critically on the flow of information, including feedback loops.

Information flow has been studied extensively by sociologist Ron Westrum, primarily in the context of accidents and human errors in aviation and healthcare. Westrum realized that safety in these contexts could be predicted by organizational culture, and developed a "continuum of safety cultures" with three categories:[6]

> *Pathological* organizations are characterized by large amounts of fear and threat. People often hoard information or withhold it for political reasons, or distort it to make themselves look better.
>
> *Bureaucratic* organizations protect departments. Those in the department want to maintain their "turf," insist on their own rules, and generally do things by the book—*their* book.

6 [westrum-2014]

> *Generative* organizations focus on the mission. How do we accomplish our goal? Everything is subordinated to good performance, to doing what we are supposed to do.

These cultures process information in different ways. Westrum observes that "the climate that provides good information flow is likely to support and encourage other kinds of cooperative and mission-enhancing behavior, such as problem solving, innovations, and interdepartmental bridging. When things go wrong, pathological climates encourage finding a scapegoat, bureaucratic organizations seek justice, and the generative organization tries to discover the basic problems with the system." The characteristics of the various types of culture are shown in Table 1-1.

Table 1-1. How organizations process information

Pathological (power-oriented)	Bureaucratic (rule-oriented)	Generative (performance-oriented)
Low cooperation	Modest cooperation	High cooperation
Messengers shot	Messengers neglected	Messengers trained
Responsibilities shirked	Narrow responsibilities	Risks are shared
Bridging discouraged	Bridging tolerated	Bridging encouraged
Failure leads to scapegoating	Failure leads to justice	Failure leads to enquiry
Novelty crushed	Novelty leads to problems	Novelty implemented

Westrum's typology has been extensively elaborated upon, and has a visceral quality that will appeal to anybody who has worked in a pathological (or even bureaucratic) organization. However, some of its implications are far from academic.

In 2013, PuppetLabs, IT Revolution Press, and ThoughtWorks surveyed 9,200 technologists worldwide to find out what made high-performing organizations successful. The resulting *2014 State of DevOps Report* is based on analysis of answers from people working in a variety of industries including finance, telecoms, retail, government, technology, education, and healthcare.[7] The headline result from the survey was that strong IT performance is a competitive advantage. Analysis showed that firms with high-performing IT organizations were

7 [forsgren]

twice as likely to exceed their profitability, market share, and productivity goals.[8]

The survey also set out to examine the cultural factors that influenced organizational performance. The most important of these turned out to be whether people were satisfied with their jobs, based on the extent to which they agreed with the following statements (which are strongly reminiscent of the reaction of the NUMMI workers who were introduced to the Toyota Production System):

- I would recommend this organization as a good place to work.
- I have the tools and resources to do my job well.
- I am satisfied with my job.
- My job makes good use of my skills and abilities.

The fact that job satisfaction was the top predictor of organizational performance demonstrates the importance of intrinsic motivation. The team working on the survey wanted to look at whether Westrum's model was a useful tool to predict organizational performance.[9] Thus the survey asked people to assess their team culture along each of the axes of Westrum's model as shown in Table 1-1, by asking them to rate the extent to which they agreed with statements such as "On my team, failure causes enquiry."[10] In this way, the survey was able to measure culture.

Statistical analysis of the results showed that team culture was not only strongly correlated with organizational performance, it was also a strong predictor of job satisfaction. The results are clear: a high-trust, generative culture is not only important for creating a safe working environment—it is the foundation of creating a high-performance organization.

Mission Command: An Alternative to Command and Control

High-trust organizational culture is often contrasted to what is popularly known as "command and control": the idea from scientific management that

8 The survey measured organizational performance by asking respondents to rate their organization's relative performance in terms of achieving its profitability, market share, and productivity goals. This is a standard scale that has been validated multiple times in prior research. See [widener].

9 In the interests of full disclosure, Jez was part of the team behind the *2014 State of DevOps Report*.

10 This method of measuring attitudes quantitatively is known as a Likert scale.

the people in charge make the plans and the people on the ground execute them—which is usually thought to be modelled on how the military functions. In reality, however, this type of command and control has not been fashionable in military circles since 1806 when the Prussian Army, a classic plan-driven organization, was decisively defeated by Napoleon's decentralized, highly motivated forces. Napoleon used a style of war known as *maneuver warfare* to defeat larger, better-trained armies. In maneuver warfare, the goal is to minimize the need for actual fighting by disrupting your enemy's ability to act cohesively through the use of shock and surprise. A key element in maneuver warfare is being able to learn, make decisions, and act faster than your enemy—the same capability that allows startups to disrupt enterprises.[11]

Three men were especially important to the reconstruction of the Prussian Army following its defeat by Napoleon: Carl von Clausewitz, David Scharnhorst, and Helmuth von Moltke. Their contributions not only transformed the military doctrine; they have important implications for people leading and managing large organizations. This particularly applies to the idea of *Auftragstaktik*, or Mission Command, which we will explore here. Mission Command is what enables maneuver warfare to work at scale—it is key to understanding how enterprises can compete with startups.

Following the eventual defeat of Napoleon, General David Scharnhorst was made Chief of the newly established Prussian General Staff. He put together a reform commission which conducted a postmortem and began to transform the Prussian Army. Scharnhorst noted that Napoleon's officers had the authority to make decisions as the situation on the ground changed, without waiting for approval through the chain of command. This allowed them to adapt rapidly to changing circumstances.

Scharnhorst wanted to develop a similar capability in a systematic way. He realized this required the training of a independent, intelligent cadre of staff officers who shared similar values and would be able to act decisively and autonomously in the heat of battle. Thus military schools were set up to train staff officers, who for the first time were accepted from all social backgrounds based on merit.

In 1857, Helmuth von Moltke, perhaps best known for his saying "no plan survives contact with the enemy," was appointed Chief of the General Staff of the Prussian Army. His key innovation, building on the military culture established by Scharnhorst, was to treat military strategy as a series of options which were to be explored extensively by officers in advance of the battle. In

11 As we discuss in Chapter 3, this concept is formalized in John Boyd's OODA (observe-orient-decide-act) loop, which in turn inspired Eric Ries' build-measure-learn loop.

1869 he issued a directive titled "Guidance for Large Unit Commanders" which sets out how to lead a large organization under conditions of uncertainty.

In this document, von Moltke notes that "in war, circumstances change very rapidly, and it is rare indeed for directions which cover a long period of time in a lot of detail to be fully carried out." He thus recommends "not commanding more than is strictly necessary, nor planning beyond the circumstances you can foresee." Instead, he has this advice: "The higher the level of command, the shorter and more general the orders should be. The next level down should add whatever further specification it feels to be necessary, and the details of execution are left to verbal instructions or perhaps a word of command. This ensures that everyone retains freedom of movement and decision within the bounds of their authority…The rule to follow is that an order should contain all, but also only, what subordinates cannot determine for themselves to achieve a particular purpose."

Crucially, orders always include a passage which describes their *intent*, communicating the *purpose* of the orders. This allows subordinates to make good decisions in the face of emerging opportunities or obstacles which prevent them from following the original orders exactly. Von Moltke notes that "there are numerous situations in which an officer must act on his own judgment. For an officer to wait for orders at times when none can be given would be quite absurd. But as a rule, it is when he acts in line with the will of his superior that he can most effectively play his part in the whole scheme of things."

These ideas form the core of the doctrine of *Auftragstaktik*, or Mission Command, which, in combination with the creation of a professionally trained cadre of staff officers who understood how to apply the doctrine operationally, was adopted by multiple elite military units, including the US Marine Corps as well as (more recently) NATO.

The history of the Prussian Army's development of *Auftragstaktik* is described in more detail in Stephen Bungay's treatise on business strategy, *The Art of Action* (from which the above quotations from "Guidance for Large Unit Commanders" are taken).[12] Bungay develops a theory of directing strategy at scale which builds on the work of Scharnhorst, von Moltke, and another Prussian general, Carl von Clausewitz. As a 26-year old, Clausewitz had fought against Napoleon in the fateful battles of Jena and Auerstadt. He subsequently served on Scharnhorst's reform commission and bequeathed us his unfinished *magnum opus*, *On War*. In this work he introduces the concept of the "fog of war"—the fundamental uncertainty we face as actors in a large and rapidly

12 [bungay]

changing environment, with necessarily incomplete knowledge of the state of the system as a whole. He also introduces the idea of *friction* which prevents reality from behaving in an ideal way. Friction exhibits itself in the form of incomplete information, unanticipated side effects, human factors such as mistakes and misunderstandings, and the accumulation of unexpected events.

Friction and Complex Adaptive Systems

Clausewitz' concept of friction is an excellent metaphor to understand the behavior of complex adaptive systems such as an enterprise (or indeed any human organization). The defining characteristic of a complex adaptive system is that its behavior at a global level cannot be understood through Taylor's reductionist approach of analyzing its component parts. Rather, many properties and behavior patterns of complex adaptive systems "emerge" from *interactions* between events and components at multiple levels within the system. In the case of open systems (such as enterprises), we also have to consider interactions with the environment, including the actions of customers and competitors, as well as wider social and technological changes.[13] Friction is ultimately a consequence of the human condition—the fact that organizations are composed of people with independent wills and limited information. Thus friction cannot be overcome.

Bungay argues that friction creates three gaps. First, a *knowledge gap* arises when we engage in planning or acting due to the necessarily imperfect state of the information we have to hand, and our need to making assumptions and interpret that information. Second, an *alignment gap* is the result of people failing to do things as planned, perhaps due to conflicting priorities, misunderstandings, or simply someone forgetting or ignoring some element of the plan. Finally, there is an *effects gap* due to unpredictable changes in the environment, perhaps caused by other actors, or unexpected side effects producing outcomes that differ from those we anticipated. These gaps are shown in Figure 1-1.

13 For those interested in different types of systems and how to make sense of them, we recommend studying Dave Snowden's Cynefin framework: *http://www.youtube.com/watch?v=N7oz366X0-8*.

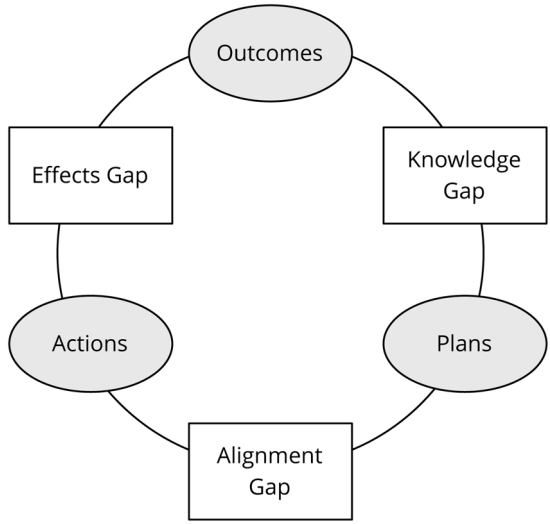

Figure 1-1. *Gaps in complex adaptive systems, from The Art of Action: How Leaders Close the Gaps between Plans, Actions, and Results by Stephen Bungay (reprinted by permission of Nicholas Brealey Publishing)*

Bungay then goes on to describe the usual scientific management remedy applied by enterprises, the alternative proposed by the doctrine of *Auftragstaktik*, and his own interpretation of Mission Command as applied to business, which he terms "directed opportunism." These are shown in Table 1-2.

Table 1-2. The three gaps, and how to manage them

	Effects gap	Knowledge gap	Alignment gap
What is it?	The difference between what we expect our actions to achieve and what they actually achieve	The difference between what we would like to know and what we actually know	The difference between what we want people to do and what they actually do
Scientific management remedy	More detailed controls	More detailed information	More detailed instructions

	Effects gap	Knowledge gap	Alignment gap
Auftragstaktik remedy	"Everyone retains freedom of decision and action within bounds"	"Do not command more than is necessary or plan beyond the circumstances you can foresee"	"Communicate to every unit as much of the higher intent as is necessary to achieve the purpose"
Directed opportunism remedy	Give individuals freedom to adjust their actions in line with intent	Limit direction to defining and communicating the intent	Allow each level to define how they will achieve the intent of the next level up, and "backbrief"

It is crucial to understand that when we work in a complex adaptive system where friction dominates, the scientific management remedies *cannot work*. In fact, they make things worse. Creating ever more detailed plans delays the feedback that would tells us which of our assumptions are invalid. Complex sets of rules and controls punish the innocent but can be evaded by the guilty, all the while destroying morale, innovation, and entrepreneurialism. Intelligence gathering fails in the face of bureaucratic or pathological organizations which hide or distort information in order to protect their turf. Organizations unable to escape the grip of scientific management are perfect targets to be disrupted by organizations that understand how to move fast at scale.

Create Alignment at Scale Following the Principle of Mission

The most important concern leaders and managers operating within a complex adaptive system face is this: how can we enable people within the organization to make good decisions—to act in the best interests of the organization—given that they can *never* have sufficient information and context to understand the full consequences of their decisions, and given that events often overtake our plans?

In *The Principles of Product Development Flow*,[14] Donald Reinertsen presents the *Principle of Mission*, based on the doctrine of Mission Command, in which we "specify the end state, its purpose, and the minimum possible constraints." According to the Principle of Mission, we create alignment not by making a detailed plan of how we achieve our objective but by describing the *intent* of our mission and communicating *why* we are undertaking it.

The key to the Principle of Mission is to create alignment and enable autonomy by setting out clear, high-level target conditions with an agreed time

14 [reinertsen]

frame—which gets smaller under conditions of greater uncertainty—and then leaving the details of *how* to achieve the conditions to teams. This approach can even be applied to multiple levels of hierarchy, with each level reducing the scope while providing more context. In the course of the book, this principle is applied in multiple contexts:

Budgeting and financial management
: Instead of a traditional budgeting process which requires all spending for the next year to be planned and locked down based on detailed projections and business plans, we set out high-level objectives across multiple perspectives such as people, organization, operations, market, and finance that are reviewed regularly. This kind of exercise can be used at multiple levels, with resources allocated dynamically when needed and the indicators reviewed on a regular basis.

Program management
: Instead of creating detailed, upfront plans on the work to be done and then breaking that work down into tiny little bits distributed to individual teams, we specify at the program level only the measurable objectives for each iteration. The teams then work out how to achieve those objectives, including collaborating with other teams and continuously integrating and testing their work to ensure they will meet the program-level objectives.

Process improvement
: Working to continuously improve processes is a key element of the TPS and a powerful tool to transform organizations. In Chapter 6 we present the Improvement Kata in which we work in iterations, specifying target objectives for processes and providing the people who operate the processes the time and resources to run experiments they need to meet the target objectives for the next iteration.

Crucially, these mission-based processes must *replace* the command and control processes, not run alongside them. This requires people to behave and act in different ways and to learn new skills. It also requires a cultural change within the organization, as we discuss in Chapter 11. Discussing how to apply Mission Command in business, Stephen Bungay reflects on a culture that enables Mission Command—which, not coincidentally, has the same characteristics that we find in the generative organizations described by Westrum in Table 1-1:

> The unchanging core is a holistic approach which affects recruiting, training, planning, and control processes, but also the culture and values of an organization. Mission Command embraces a conception of leadership which unsentimentally places human beings at its center. It crucially depends on factors which do not appear on the balance sheet

> of an organization: the willingness of people to accept responsibility; the readiness of their superiors to back up their decisions; the tolerance of mistakes made in good faith. Designed for an external environment which is unpredictable and hostile, it builds on an internal environment which is predictable and supportive. At its heart is a network of trust binding people together up, down, and across a hierarchy. Achieving and maintaining that requires constant work.[15]

Your People Are Your Competitive Advantage

The story of the Fremont Assembly site doesn't stop with NUMMI. It is in fact the locus of two paradigm shifts in the US auto manufacturing industry. In 2010, the NUMMI plant was purchased by Tesla Motors and became the Tesla Factory. Tesla uses continuous methods to innovate faster than Toyota, discarding the concept of model years in favor of more frequent updates and in many cases enabling owners of older cars to download new firmware to gain access to new features. Tesla has also championed transparency of information, announcing it will not enforce its patents. In doing so, it echoes a story from Toyota's origins when it used to build automatic looms. Upon hearing that the plans for one of the looms had been stolen, Kiichiro Toyoda is said to have remarked:

> Certainly the thieves may be able to follow the design plans and produce a loom. But we are modifying and improving our looms every day. So by the time the thieves have produced a loom from the plans they stole, we will have already advanced well beyond that point. And *because they do not have the expertise gained from the failures it took to produce the original*, they will waste a great deal more time than us as they move to improve their loom. We need not be concerned about what happened. We need only continue as always, making our improvements.[16]

The long-term value of an enterprise is not captured by the value of its products and intellectual property but rather by its ability to continuously increase the value it provides to customers—and to create *new* customers—through innovation.

A key premise of this book—supported by the experience of companies such as Tesla, among many, many others—is that the flexibility provided by software can, when correctly leveraged, accelerate the innovation cycle. Software can

15 [bungay], p. 88.

16 [rother-2010], p. 40, emphasis ours.

provide your enterprise with a competitive advantage by enabling you to search for new opportunities and execute validated opportunities faster than the competition. The good news is that these capabilities are within the reach of all enterprises, not just tech giants. The data from the *2014 State of Devops Report* shows that 20% of organizations with more than 10,000 employees fall into the high-performing group—a smaller percentage than smaller companies, but still significant.

Many people working in enterprises believe that there is some essential difference between them and tech giants such as Google, Amazon, or Netflix that are held up as examples of technology "done right." We often hear, "that won't work here." That may be right, but people often look in the wrong places for the obstacles that prevent them from improving. Skeptics often treat size, regulation, perceived complexity, legacy technology, or some other special characteristic of the domain in which they operate as a barrier to change. The purpose of this chapter is to show that while these obstacles are indeed challenges, the most serious barrier is to be found in organizational culture, leadership, and strategy.

Many organizations try to take shortcuts to higher performance by starting innovation labs, acquiring startups, adopting methodologies, or reorganizing. But such efforts are neither necessary nor sufficient. They can only succeed if combined with efforts to create a generative culture and strategy across the whole organization, including suppliers—and if this is achieved, there will be no need to resort to such shortcuts.

The second chapter of this book describes the principles that enable organizations to succeed in the long term by balancing their portfolio of products. In particular, we distinguish two independent types of activity in the product development lifecycle: *exploring* new ideas to gather data and eliminate those that will not see rapid uptake by users, and *exploiting* those that we have validated against the market. Part II of the book discusses how to run the explore domain, with Part III covering the exploit domain. Finally, Part IV of the book shows how to transform your organization focusing on culture, financial management, governance, risk, and compliance.

PART III

EXPLOIT

> Prediction is difficult, especially about the future.
>
> — Niels Bohr

In Part II, we showed how to explore new opportunities—whether potential products or internal tools and services. In this part, we discuss how to exploit validated ideas. As discussed in Chapter 2, these two domains demand a completely different approach for management and execution. However, both are necessary—and indeed complementary—if we are to effectively balance our enterprise portfolio and adapt to a constantly changing business environment.

We hope you are reading this part because you have successfully exited the explore domain—but it's just as likely that you are here because you participate in a large program of work in an enterprise which has been set up in the traditional way. Thus, this part of the book primarily describes how to change the way we lead and manage such large-scale programs of work in a way that empowers employees and dramatically increases the rate at which we can deliver valuable, high-quality products to customers. But before we can begin, we must understand our current condition.

In an enterprise context, planned work is usually prioritized through a centralized or departmental planning and budgeting process. Approved projects then go through the development process before going live or being released to manufacturing. Even in organizations which have adopted "agile" development methods, the value stream required to deliver a project often resembles

Figure III-1, which we describe as "water-scrum-fall."[1] In cases where one or more of these phases are outsourced, we must also go through a procurement process before we can proceed to the design and development phases following approval. Because this process is so onerous, we tend to batch up work, creating large programs which further exacerbate the problems with the project paradigm.

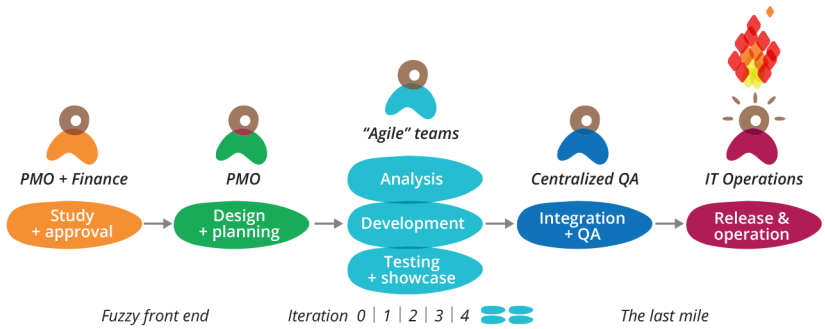

Figure III-1. *Water-scrum-fall*

This project-based paradigm for carrying out software development at scale has its origins in the post-WWII United States military-industrial complex, where software was crucial for building a new generation of airplanes, missile systems, and spacecraft that had essentially one customer: the US government. It's no coincidence that the term "software engineering" was coined at a 1968 NATO conference which was convened to work out how to formalize large-scale software development.[2]

The traditional centralized phase-gate project paradigm was designed in a simpler era. Products could not deliver value until they were fully manufactured, they didn't need to change substantially over the course of their lifecycle, and we had a high level of confidence that we would not need to change the specification significantly in response to new information discovered in the course of building the product.

None of these criteria apply to software-based systems today, and the power of software derives from the fact that it's cheap to prototype and change. In particular, since we are so frequently wrong about what users of our products and systems will find valuable, planning out big programs of work months in

1 The term "water-scrum-fall" was coined by Forrester Research. A *value stream* is defined as "the sequence of activities an organization undertakes to deliver on a customer request" [martin], p. 2. We cover value streams in Chapter 7.

2 *http://homepages.cs.ncl.ac.uk/brian.randell/NATO*

advance leads to an enormous amount of waste and bad blood. Instead of trying to get better at predicting the future, we should improve our ability to adapt rapidly and effectively to new information.[3]

The modern, *lean-agile* paradigm we present for running large-scale programs of work discussed in this part is the result of working with and studying a number of organizations that need to get software development off the critical path. They want to move fast at scale, detect weak signals in the market, and exploit them rapidly. This is what allows them to provide better customer service, to reduce the cost of creating and evolving products, and to increase quality and stability of their services.

There are several frameworks that deal with scaling agile software development methods. In general, these frameworks take small teams practicing Scrum and add more structures on top to coordinate their work. However, these teams are still embedded within a phase-gate program and portfolio management process that is more or less unchanged from the traditional project management paradigm. They still apply top-down thinking and tend to batch up work into releases with long cycle times, thus limiting the use of the information collected to guide future decisions. Our approach differs in several important respects from these frameworks, as well as from more traditional phase-gate frameworks.

The most important difference is that instead of presenting a particular set of processes and practices to implement, we focus on implementing continuous improvement at the senior leadership level to drive the *evolution* of your organization and the processes you use. Continuous improvement cannot be at the edges of our "big diagram": we put it front and center. This reflects the fact that there is no one-size-fits-all solution and that every organization faces a different set of circumstances. Every organization will take its own path to address changes, aligned to its own business objectives; to create lasting results, we must enable teams to try things out and learn what works and what doesn't for themselves.

In the following chapters, we will present the following principles for lean-agile product development at scale:

- Implement an iterative continuous improvement process at the leadership level with concise, clearly specified outcomes to create alignment at scale, following the Principle of Mission.
- Work scientifically towards challenging goals, which will lead you to identifying and removing—or avoiding—no-value-add activity.

3 This is a key claim of Nassim Taleb's body of work.

- Use continuous delivery to reduce the risk of releases, decrease cycle time, and make it economic to work in small batches.
- Evolve an architecture that supports loosely coupled customer-facing teams which have autonomy in *how* they work to achieve the program-level outcomes.
- Reduce batch sizes and take an experimental approach to the product development process.
- Increase and amplify feedback loops to make smaller, more frequent decisions based on the information we learn from performing our work to maximize customer value.

We'll also provide several examples of enterprises that have leveraged these principles to create a lasting competitive advantage, and describe how they transformed themselves in the process.

CHAPTER 6

Deploy Continuous Improvement

> The paradox is that when managers focus on productivity, long-term improvements are rarely made. On the other hand, when managers focus on quality, productivity improves continuously.
>
> — John Seddon

In most enterprises, there is a distinction between the people who build and run software systems (often referred to as "IT") and those who decide what the software should do and make the investment decisions (often called "the business"). These names are relics of a bygone age in which IT was considered a cost necessary to improve efficiencies of the business, not a creator of value for external customers by building products and services. These names and the functional separation have stuck in many organizations (as has the relationship between them, and the mindset that often goes with the relationship). Ultimately, we aim to remove this distinction. In high-performance organizations today, people who design, build, and run software-based products are an integral part of business; they are given—and accept—responsibility for customer outcomes. But getting to this state is hard, and it's all too easy to slip back into the old ways of doing things.

Achieving high performance in organizations that treat software as a strategic advantage relies on *alignment* between the IT function and the rest of the organization, along with the ability of IT to *execute*. It pays off. In a report for the *MIT Sloan Management Review*, "Avoiding the Alignment Trap in Information Technology," the authors surveyed 452 companies and discovered that

high performers (7% of the total) spent a little less than average on IT while achieving substantially higher rates of revenue growth.[1]

However, *how* you move from low performance to high performance matters. Companies with poor alignment and ineffective IT have a choice. Should they pursue alignment first, or try to improve their ability to execute? The data shows that companies whose IT capabilities were poor achieve worse results when they pursue alignment with business priorities before execution, even when they put significant additional investment into aligned work. In contrast, companies whose engineering teams do a good job of delivering their work on schedule and simplifying their systems achieve better business results with much lower cost bases, even if their IT investments aren't aligned with business priorities.

The researchers concluded that to achieve high performance, companies that rely on software should focus first and foremost on their ability to execute, build reliable systems, and work to continually reduce complexity. Only then will pursuing alignment with business priorities pay off.

However, in every team we are always balancing the work we do to improve our capability against delivery work that provides value to customers. In order to do this effectively, it's essential to manage both kinds of work at the program and value stream levels. In this chapter we describe how to achieve this by putting in place a framework called *Improvement Kata*. This is the first step we must take to drive continuous improvement in our execution of large scale programs. Once we have achieved this, we can use the tools in the following chapters to identify and remove no-value-add activity in our product development process.

The HP LaserJet Firmware Case Study

We will begin with a case study from the HP LaserJet Firmware team, which faced a problem with both alignment and execution.[2] As the name suggests, this was a team working on embedded software, whose customers have no desire to receive software updates frequently. However, it provides an excellent example of how the principles described in the rest of Part III work at scale in a distributed team, as well as of the economic benefits of adopting them.

HP's LaserJet Firmware division builds the firmware that runs all their scanners, printers, and multifunction devices. The team consists of 400 people distributed across the USA, Brazil, and India. In 2008, the division had a

1 [schpilberg]

2 This case study is taken from [gruver], supplemented by numerous discussions with Gary Gruver.

problem: they were moving too slowly. They had been on the critical path for all new product releases for years, and were unable to deliver new features: "Marketing would come to us with a million ideas that would dazzle the customer, and we'd just tell them, 'Out of your list, pick the two things you'd like to get in the next 6–12 months.'" They had tried spending, hiring, and outsourcing their way out of the problem, but nothing had worked. They needed a fresh approach.

Their first step was to understand their problem in greater depth. They approached this by using *activity accounting*—allocating costs to the activities the team is performing. Table 6-1 shows what they discovered.

Table 6-1. Activities of the HP LaserJet Firmware team in 2008

% of costs	Activity
10%	Code integration
20%	Detailed planning
25%	Porting code between version control branches
25%	Product support
15%	Manual testing
~5%	Innovation

This revealed a great deal of no-value-add activity in their work, such as porting code between branches and detailed upfront planning. The large amount spent on current product support also indicated a problem with the quality of the software being produced. Money spent on support is generally serving *failure demand*, as distinct from *value demand* was only driving 5% of the team's costs.[3]

The team had a goal of increasing the proportion of spending on innovation by a factor of 10. In order to achieve that goal, they took the bold but risky decision to build a new firmware platform from scratch. There were two main architectural goals for the new "FutureSmart" platform. The first goal was to increase quality while reducing the amount of manual testing required for new

3 The distinction between failure demand and value demand comes from John Seddon, who noticed that when banks outsourced their customer service to call centers, the volume of calls rose enormously. He showed that up to 80% of the calls were "failure demand" of people calling back because their problems were not solved correctly the first time [seddon].

firmware releases (a full manual testing cycle required six weeks). The team hoped that this goal could be achieved through:

- The practice of continuous integration (which we describe in Chapter 8)
- Significant investment in test automation
- Creating a hardware simulator so that tests could be run on a virtual platform
- Reproduction of test failures on developer workstations

Three years into the development of the new firmware, thousands of automated tests had been created.

Second, they wanted to remove the need for the team to spend time porting code between branches (25% of total costs on the existing system). This was caused by the need to create a branch—effectively a copy of the entire codebase—for every new line of devices under development. If a feature or bug-fix added to one line of devices was required for any others, these changes would need to be merged (copied back) into the relevant code branches for the target devices, as shown in Figure 6-1. Moving away from branch-based development to trunk-based development was also necessary to implement continuous integration. Thus the team decided to create a single, modular platform that could run on any device, removing the need to use version control branches to handle the differences between devices.

The final goal of the team was to reduce the amount of time its members spent on detailed planning activities. The divisions responsible for marketing the various product lines had insisted on detailed planning because they simply could not trust the firmware team to deliver. Much of this time was spent performing detailed re-plans after failing to meet the original plans.

Furthermore, the team did not know how to implement the new architecture, and had not used trunk-based development or continuous integration at scale before. They also understood that test automation would require a great deal of investment. How would they move forward?

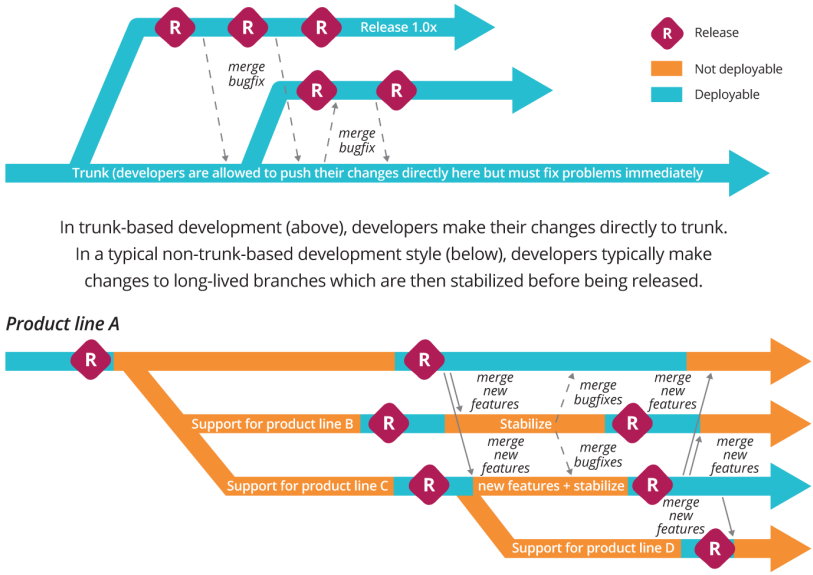

Figure 6-1. *Branching versus trunk-based development*

It's all too easy to turn a sequence of events into a story in an attempt to explain the outcome—a cognitive bias that Nassim Taleb terms the *narrative fallacy*. This is, arguably, how methodologies are born. What struck us when studying the FutureSmart case were the similarities between the program management method of FutureSmart's engineering management team and the approach Toyota uses to manage innovation as described in Mike Rother's *Toyota Kata: Managing People for Improvement, Adaptiveness, and Superior Results*.[4]

Drive Down Costs Through Continuous Process Innovation Using the Improvement Kata

The Improvement Kata, as described by Mike Rother, is a general-purpose framework and a set of practice routines for reaching goals where the path to the goal is uncertain. It requires us to proceed by iterative, incremental steps, using very rapid cycles of experimentation. Following the Improvement Kata also increases the capabilities and skills of the people doing the work, because it requires them to solve their own problems through a process of continuous experimentation, thus forming an integral part of any learning organization.

4 [rother-2010]

Finally, it drives down costs through identifying and eliminating waste in our processes.

The Improvement Kata needs to be first adopted by the organization's management, because it is a management philosophy that focuses on developing the capabilities of those they manage, as well as on enabling the organization to move towards its goals under conditions of uncertainty. Eventually, everybody in the organization should be practicing the Improvement Kata habitually to achieve goals and meet challenges. This is what creates a culture of continuous improvement, experimentation, and innovation.

To understand how this works, let's examine the concept of *kata* first. A kata is "a routine you practice deliberately, so its pattern becomes a habit."[5] Think of practicing scales to develop muscle memory and digital dexterity when learning the piano, or practicing the basic patterns of movement when learning a martial art (from which the term derives), or a sport. We want to make continuous improvement a habit, so that when faced with an environment in which the path to our goal is uncertain, we have an instinctive, unconscious routine to guide our behavior.

In Toyota, one of the main tasks of managers is to teach the Improvement Kata pattern to their teams and to facilitate running it (including coaching learners) as part of everyday work. This equips teams with a method to solve their own problems. The beauty of this approach is that if the goal or our organization's environment changes, we don't need to change the way we work—if everybody is practicing the Improvement Kata, the organization will automatically adapt to the new conditions.

The Improvement Kata has four stages that we repeat in a cycle, as shown in Figure 6-2.

5 [rother]

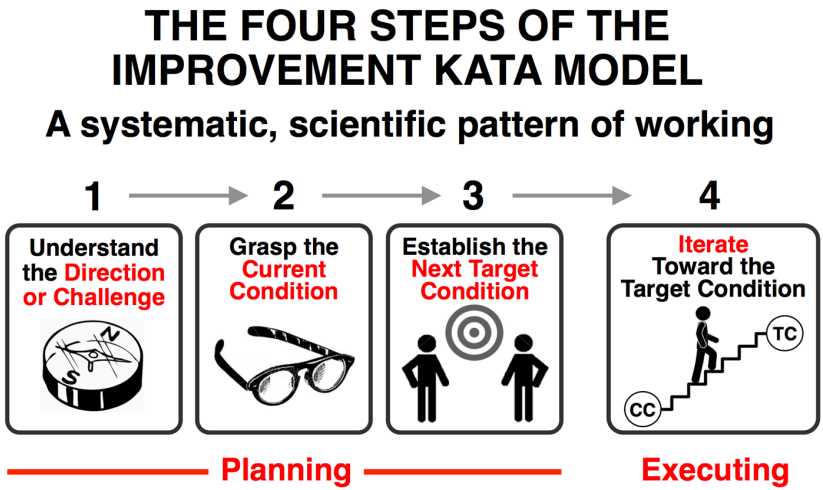

Figure 6-2. *The Improvement Kata, courtesy of Mike Rother*

Understand the Direction

We begin with understanding the direction. Direction is derived from the vision set by the organization's leadership. A good vision is one that is inspiring—and, potentially, unattainable in practice. For example, the long-term vision for Toyota's production operations is "One-piece flow at lowest possible cost." In *Leading Lean Software Development*, Mary and Tom Poppendieck describe Paul O'Neill setting the objective for Alcoa to be "Perfect safety for all people who have anything to do with Alcoa" when he became CEO in 1987.[6]

People need to understand that they must always be working towards the vision and that they will never be done with improvement. We will encounter problems as we move towards the vision. The trick is to treat them as obstacles to be removed through experimentation, rather than objections to experimentation and change.

Based on our vision and following the Principle of Mission, we must understand the direction we are working in, at the level of the whole organization and at the value stream level. This challenge could be represented in the form of a future-state value stream map (see Chapter 7 for more on value stream mapping). It should result in a measurable outcome for our customers, and we should plan to achieve it in six months to three years.

6 [poppendieck-09], Frame 13, "Visualize Perfection."

Planning: Grasp the Current Condition and Establish a Target Condition

After we have understood the direction at the organizational and value stream levels, we incrementally and iteratively move towards it at the process level. Rother recommends setting target conditions with a horizon between one week and three months out, with a preference for shorter horizons for beginners. For teams that are using iterative, incremental methods to perform product development, it makes sense to use the same iteration (or sprint) boundaries for both product development and Improvement Kata iterations. Teams that use flow-based methods such as the Kanban Method (for which see Chapter 7) and continuous delivery (described in Chapter 8) can create Improvement Kata iterations at the program level.

As with all iterative product development methods, Improvement Kata iterations involve a planning part and an execution part. Here, planning involves grasping the current condition at the process level and setting a target condition that we aim to achieve by the end of the next iteration.

Analyzing the current condition "is done to obtain the facts and data you need in order to then describe an appropriate next target condition. What you're doing is trying to find the current pattern of operation, so you can establish a desired pattern of operation (a target condition)." The target condition "describes in measurable detail how you want a process to function…[It is] a description and specification of the operating pattern you want a process or system to have on a future date."[7]

The team grasps the current condition and establishes a target condition together. However, in the planning phase the team does not plan how to *move* to the target condition. In the Improvement Kata, people doing the work strive to achieve the target condition by performing a series of experiments, not by following a plan.

A target condition identifies the process being addressed, sets the date by which we aim to achieve the specified condition, and specifies measurable details of the process as we want it to exist. Examples of target conditions include WIP (work in progress) limits, the implementation of Kanban or a continuous integration process, the number of good builds we expect to get per day, and so forth.

7 [rother]

Getting to the Target Condition

Since we are engaging in process innovation in conditions of uncertainty, we cannot know in advance how we will achieve the target condition. It's up to the people doing the work to run a series of experiments using the Deming cycle (plan, do, check, act), as described in Chapter 3. The main mistakes people make when following the Deming cycle are performing it too infrequently and taking too long to complete a cycle. With Improvement Kata, everybody should be running experiments on a daily basis.

Each day, people in the team go through answering the following five questions:[8]

1. What is the target condition?
2. What is the actual condition now?
3. What obstacles do you think are preventing you from reaching the target condition? Which one are you addressing now?
4. What is your next step? (Start of PDCA cycle.) What do you expect?
5. When can we go and see what we learned from taking that step?

As we continuously repeat the cycle, we reflect on the last step taken to introduce improvement. What did we expect? What actually happened? What did we learn? We might work on the same obstacle for several days.

This experimental approach is already central to how engineers and designers work. Designers who create and test prototypes to reduce the time taken by a user to complete a task are engaged in exactly this process. For software developers using test-driven development, every line of production code they write is essentially part of an experiment to try and make a unit test pass. This, in turn, is a step on the way to improving the value provided by a program—which can be specified in the form of a target condition, as we describe in Chapter 9.

The Improvement Kata is simply a generalization of this approach to improvement, combined with applying it at multiple levels of the organization, as we discuss when presenting strategy deployment in Chapter 15.

How the Improvement Kata Differs from Other Methodologies

You can think of the Improvement Kata as a *meta-methodology* since it does not apply to any particular domain, nor does it tell you what to do. It is not a playbook; rather, as with the Kanban Method, it teaches teams how to *evolve*

8 [rother]

their existing playbook. In this sense, it differs from other agile frameworks and methodologies. With the Improvement Kata, there is no need to make existing processes conform to those specified in the framework; process and practices you use are *expected* to evolve over time. *This* is the essence of agile: teams do not become agile by adopting a methodology. Rather, true agility means that teams are constantly working to evolve their processes to deal with the particular obstacles they are facing at any given time.

NOTE

Single-Loop Learning and Double-Loop Learning

Changing the way we think and behave in reaction to a failure is crucial to effective learning. This is what distinguishes *single-loop learning* from *double-loop learning* (see Figure 6-3). These terms were coined by management theorist Chris Argyris, who summarizes them as follows: "When the error detected and corrected permits the organization to carry on its present policies or achieve its present objectives, then that error-and-correction process is single-loop learning. Single-loop learning is like a thermostat that learns when it is too hot or too cold and turns the heat on or off. The thermostat can perform this task because it can receive information (the temperature of the room) and take corrective action. Double-loop learning occurs when error is detected and corrected in ways that involve the modification of an organization's underlying norms, policies and objectives."[9] Argyris argues that the main barrier to double-loop learning is defensiveness when confronted with evidence that we need to change our thinking, which can operate at both individual and organizational levels. We discuss how to overcome this anxiety and defensiveness in Chapter 11.

9 [argyris], pp. 2–3.

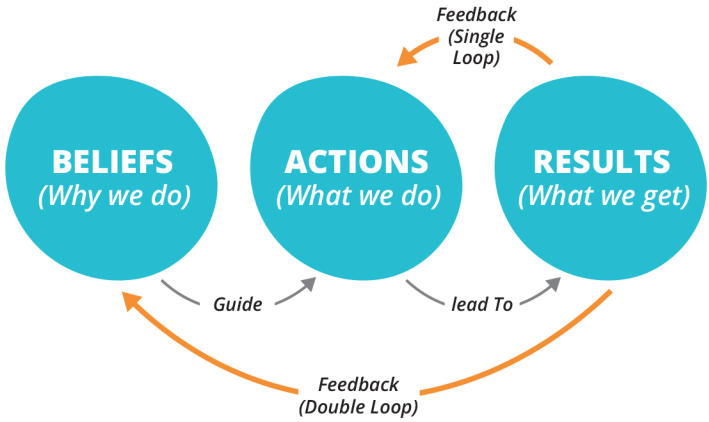

Figure 6-3. *Single-loop and double-loop learning*

When you practice the Improvement Kata, process improvement becomes planned work, similar to building product increments. The key is that we don't plan *how* we will achieve the target condition, nor do we create epics, features, stories, or tasks. Rather, the team works this out through experimentation over the course of an iteration.

Deploying the Improvement Kata

Rother's work on the Improvement Kata was a direct result of his enquiry into how people become managers at Toyota. There is no formal training program, nor is there any explicit instruction. However, to become a manager at Toyota, one must have first worked on the shop floor and therefore participated in the Improvement Kata. Through this process, managers receive implicit training in how to manage at Toyota.

This presents a problem for people who want to learn to manage in this way or adopt the Improvement Kata pattern. It is also a problem for Toyota—which is aiming to scale faster than is possible through what is effectively an apprenticeship model for managers.

Consequently, Rother presents the Coaching Kata in addition to the Improvement Kata. It is part of deploying the Improvement Kata, but it is also as a way to grow people capable of working with the Improvement Kata, including managers.

Rother has made a guide to deploying the Improvement Kata, *The Improvement Kata Handbook*, available for free on his website at http://bit.ly/11iBzlY.

How the HP LaserJet Team Implemented the Improvement Kata

The direction set by the HP LaserJet leadership was to improve developer productivity by a factor of 10, so as to get firmware off the critical path for product development and reduce costs.[10] They had three high-level goals:

1. Create a single platform to support all devices
2. Increase quality and reduce the amount of stabilization required prior to release
3. Reduce the amount of time spent on planning

They did not know the details of the path to these goals and didn't try to define it. The key decision was to work in iterations, and set target conditions for the end of each four-week iteration. The target conditions for Iteration 30 (about 2.5 years into the development of the FutureSmart platform) are shown in Figure 6-4.

The first thing to observe is that the target conditions (or "exit criteria" as they are known in FutureSmart) are all measurable conditions. Indeed, they fulfill all the elements of SMART objectives: they are specific, measurable, achievable, relevant, and time bound (the latter by virtue of the iterative process). Furthermore, many of the target conditions were not focused on features to be delivered but on attributes of the system, such as quality, and on activities designed to validate these attributes, such as automated tests. Finally, the objectives for the entire 400-person distributed program for a single month was captured in a concise form that fit on a single piece of paper—similar to the standard A3 method used in the Toyota Production System.

How are the target conditions chosen? They are "aggressive goals the team feels are possible and important to achieve in 4 weeks…We typically drive hard for these stretch goals but usually end up hitting around 80% of what we thought we could at the beginning of the month."[11] Often, target conditions would be changed or even dropped if the team found that the attempt to achieve them results in unintended consequences: "It's surprising what you learn in a month and have to adjust based on discovery in development."[12]

10 [gruver], p. 144.

11 [gruver], p. 40.

12 Ibid.

Rank	Theme	Exit Criteria Objective met / *Objective not met*
0	Quality threshold	P1 issues open < 1 week L2 test failure 24 hour response
1	Quarterly bit release	*A) Final P1 change requests fixed* B) Reliability error rate at release criteria
2	New platform stability and test coverage	A) Customer Acceptance Test 100% passing B) All L2 test pillars 98% passing C) L4 test pillars in place D) L4 test coverage for all Product Turn On requirements E) 100% execution of L4 tests on new products
3	Product Turn On dependencies and key features	A) Print for an hour at speed to finisher with stapling B) Copy for an hour *at speed* *C) Enable powersave mode* D) Manufacturing nightly test suite execution E) Common Test Library support for four-line control panel display
4	Build for next-gen products	*A) End-to-end system build on new processor* *B) High-level performance analysis on new processor*
5	Fleet integration plan	Align on content and schedule for "slivers" of end-to-end agile test with system test lab

Figure 6-4. *Target conditions for iteration 30*[13]

13 Gruver, Gary, Young, Mike, Fulghum, Pat. *A Practical Approach to Large-Scale Agile Development: How HP Transformed LaserJet FutureSmart Firmware*, 1st Edition, (c) 2013. Reprinted by permission of Pearson Education, Inc. Upper Saddle River, NJ.

WARNING

What Happens When We Do Not Achieve Our Target Conditions?

In bureaucratic or pathological organizational cultures, not achieving 100% of the specified target conditions is typically considered a failure. In a generative culture, however, we *expect* to not be able to achieve all our target conditions. The purpose of setting aggressive target conditions is to reveal obstacles so we can overcome them through further improvement work. Every iteration should end with a retrospective (described in Chapter 11) in which we investigate how we can get better. The results form part of the input for the next iteration's target conditions. For example, if we fail to achieve a target condition for the number of good builds of the system per day, we may find that the problem is that it takes too long to provision test environments. We may then set a target condition to reduce this in the next iteration.

This approach is a common thread running through all of Lean Thinking. The subtitle of Mary and Tom Poppendieck's book *Leading Lean Software Development* reads: "Results are not the point." This is a provocative statement that gets to the heart of the lean mindset. If we achieve the results by ignoring the process, we do not learn how to improve the process. If we do not improve the process, we cannot repeatably achieve better results. Organizations that put in place unmodifiable processes that everybody is required to follow, but which get bypassed in a crisis situation, fail on both counts.

This adaptive, iterative approach is not new. Indeed it has a great deal in common with what Tom Gilb proposed in his 1988 work *Principles of Software Engineering Management*:[14]

> We must set measurable objectives for each next small delivery step. Even these are subject to constant modification as we learn about reality. It is simply not possible to set an ambitious set of multiple quality, resource, and functional objectives, and be sure of meeting them all as planned. We must be prepared for compromise and trade-off. We must then design (engineer) the immediate technical solution, build it, test it, deliver it—and get feedback. This feedback must be used to modify the immediate design (if necessary), modify the major architectural ideas (if necessary), and modify both the short-term and the long-term objectives (if necessary).

14 [gilb-88], p. 91.

Designing for Iterative Development

In large programs, demonstrating improvement within an iteration requires ingenuity and discipline. It's common to feel we can't possibly show significant progress within 2–4 weeks. Always try to find something small to bite off to achieve a little bit of improvement, instead of trying to do something you think will have more impact but will take longer.

This is not a new idea, of course. Great teams have been working this way for decades. One high-profile example is the Apple Macintosh project where a team of about 100 people—co-located in a single building—designed the hardware, operating system, and applications for what was to become Apple's breakthrough product.

The teams would frequently integrate hardware, operating system, and software to show progress. The hardware designer, Burrell Smith, employed programmable logic chips (PALs) so he could prototype different approaches to hardware design rapidly in the process of developing the system, delaying the point at which it became fixed—a great example of the use of optionality to delay making final decisions.[15]

After two years of development, the new firmware platform, FutureSmart, was launched. As a result, HP had *evolved* a set of processes and tools that substantially reduced the cost of no-value-add activities in the delivery process while significantly increasing productivity. The team was able to achieve "predictable, on-time, regular releases so new products could be launched on time."[16] Firmware moved off the critical path for new product releases for the first time in twenty years. This, in turn, enabled them to build up trust with the product marketing department.

As a result of the new relationship between product marketing and the firmware division, the FutureSmart team was able to considerably reduce the time spent on planning. Instead of "committing to a final feature list 12 months in advance that we could never deliver due to all the plan changes over the time,"[17] they looked at each planned initiative once every 6 months and did a 10-minute estimate of the number of months of engineering effort required for a given initiative, broken down by team. More detailed analysis would be performed once work was scheduled into an iteration or mini-milestone. An example of the output from one of these exercises is shown in Figure 6-5.

15 *http://folklore.org/StoryView.py?story=Macintosh_Prototypes.txt*

16 [gruver], p. 89.

17 [gruver], p. 67.

High-Level Estimate – FW Engineering Months

Rank	Initiative	*Component 1* (25-30)	*Component 2* (20-25)	*Component 3* (30-40)	*Component 4* (30-40)	*Component 5* (20-30)	*Component 6* (20-30)	*Component 7* (20-30)	*Component 8* (15-25)	*Component 10* (40-50)	*Component 11* (20-30)	*Component 12* (20-30)	*other teams*	**TOTAL**
1	Initiative A			*21*			*5*	*3*		*1*				***30***
2	Initiative B	*3*							*4*				*17*	***24***
3	Initiative C		*5*							*2*	*1*	*1*		***9***
4	Initiative D							*10*		*2*	*2*	*2*		***16***
5	Initiative E					*20*						*3*	*5*	***28***
6	Initiative F	*23*							*5*	*6*			*2*	***36***
7	Initiative G									*2*				***2***
8	Initiative H											*5*		***5***
9	Initiative I												*3*	***3***
10	Initiative J		*20*	*27*			*17*			*39*	*17*	*21*	*9*	***150***
11	Initiative K			*3*	*30*		*3*		*3*	*14*			*12*	***65***
12	Initiative L									*2*				***2***
13	Initiative M	*3*						*10*		*6*	*6*	*6*		***31***
		29	*25*	*51*	*30*	*20*	*25*	*23*	*12*	*74*	*26*	*38*	*59*	***401***

Figure 6-5. *Ballpark estimation of upcoming initiatives*[18]

This is significantly different from how work is planned and estimated in large projects that often create detailed functional and architectural epics which must be broken down into smaller and smaller pieces, analyzed in detail, estimated, and placed into a prioritized backlog *before* they are accepted into development.

Ultimately the most important test of the planning process is whether we are able to keep the commitments we make to our stakeholders, including end users. As we saw, a more lightweight planning process resulted in firmware development moving off the critical path, while at the same time reducing both development costs and failure demand. Since we would expect failure demand to *increase* as we increase throughput, this is doubly impressive.

18 Gruver, Gary, Young, Mike, Fulghum, Pat. *A Practical Approach to Large-Scale Agile Development: How HP Transformed LaserJet FutureSmart Firmware*, 1st Edition, (c) 2013. Reprinted by permission of Pearson Education, Inc. Upper Saddle River, NJ.

Three years after their initial measurements, a second activity-accounting exercise offered a snapshot of the results the FutureSmart team had achieved with their approach, shown in Table 6-2.

Table 6-2. Activity of the HP LaserJet Firmware Team in 2011

% of costs	Activity	Previously
2%	Continuous integration	10%
5%	Agile planning	20%
15%	One main branch	25%
10%	Product support	25%
5%	Manual testing	15%
23%	Creating and maintaining automated test suites	0%
~40%	Innovation	~5%

Overall, the HP LaserJet Firmware division changed the economics of the software delivery process by adopting continuous delivery, comprehensive test automation, an iterative and adaptive approach to program management, and a more agile planning process.

Economic Benefits of HP FutureSmart's Agile Transformation

- Overall development costs were reduced by ~40%.
- Programs under development increased by ~140%.
- Development costs per program went down 78%.
- Resources driving innovation increased eightfold.

The most important point to remember from this case study is that the enormous cost savings and improvements in productivity were only possible on the basis of a large and ongoing *investment* made by the team in test automation and continuous integration. Even today, many people think that Lean is a management-led activity and that it's about simply *cutting costs*. In reality, it requires *investing* to remove waste and reduce failure demand—it is a worker-led activity that, ultimately, can continuously drive down costs and improve quality and productivity.

Managing Demand

Up to now, we've been discussing how to improve the throughput and quality of the delivery process. However, it is very common for this kind of improvement work to get crowded out by business demands, such as developing new features. This is ironic, given that the whole purpose of improvement work is to increase the rate at which we can deliver as well as the quality of what gets delivered. It's often hard to make the outcome of improvement work tangible—which is why it's important to make it visible by activity accounting, including measuring the cycle time and the time spent serving failure demand such as rework.

The solution is to use the same mechanism to manage both demand and improvement work. One of the benefits of using the Improvement Kata approach is that it creates alignment to the outcomes we wish to achieve over the next iteration across the whole program. In the original Improvement Kata, the target conditions are concerned with process improvement, but we can use them to manage demand as well.

There are two ways to do this. In organizations with a generative culture (see Chapter 1), we can simply specify the desired business goals as target conditions, let the teams come up with ideas for features, and run experiments to measure whether they will have the desired impact. We describe how to use impact mapping and hypothesis-driven development to achieve this in Chapter 9. However, more traditional enterprises will typically have a backlog of work prioritized at the program level by its lines of business or by product owners.

We can take a few different approaches to integrating a program-level backlog with the Improvement Kata. One possibility is for teams working within the program to deploy the Kanban Method, as described in Chapter 7. This includes the specification of work in process (WIP) limits which are owned and managed by these teams. New work will only be accepted when existing work is completed (where "completed" means it is at least integrated, fully tested with all test automation completed, and shown to be deployable).

TIP

Managing Cross-Cutting Work

Implementing some features within a program will involve multiple teams working together. To achieve this, the HP FutureSmart division would set up a small, temporary "virtual" feature team whose job is to coordinate work across the relevant teams.

The HP FutureSmart program, some of whose teams were using Scrum, took the approach of specifying a target velocity at the *program* level. Work adding up to the target velocity was accepted for each iteration, approximating a WIP limit. In order to implement this approach, all work was analyzed and estimated at a high level before being accepted. Analysis and estimation was kept to the bare minimum required to be able to consistently meet the overall program-level target conditions, as shown in Figure 6-5.

WARNING

Do Not Use Team Velocity Outside Teams

It is important to note that specifying a target velocity at the program level does *not* require that we attempt to measure or manage velocity at the team level, or that teams must use Scrum. Program-level velocity specifies the expected work capacity of all teams based on high-level estimates, as shown in Figure 6-5. If a team using Scrum accepts work based on these high-level feature specifications, they then create lower-level stories with which to work.

Scrum's team-level velocity measure is not all that meaningful outside of the context of a particular team. Managers should *never* attempt to compare velocities of different teams or aggregate estimates across teams. Unfortunately, we have seen team velocity used as a measure to compare productivity between teams, a task for which it is neither designed nor suited. Such an approach may lead teams to "game" the metric, and even to stop collaborating effectively with each other. In any case, it doesn't matter how many stories we complete if we don't achieve the business outcomes we set out to achieve in the form of program-level target conditions.

In this and the next chapter, we describe a much more effective way to measure progress and manage productivity—one that does not require all teams to use Scrum or "standardize" estimates or velocity. We use activity accounting and value stream mapping (described in Chapter 7) to measure productivity, and we use value stream mapping combined with the Improvement Kata to increase it—crucially, at the value stream level rather than at the level of individual teams. We measure and manage progress through the use of target conditions at the program level, and if we need to increase visibility, we reduce the duration of iterations.

Creating an Agile Enterprise

Many organizations look to try and adopt agile methods to improve the productivity of their teams. However, agile methods were originally designed around small, cross-functional teams, and many organizations have struggled to use these methods at scale. Some frameworks for scaling agile focus on creating such small teams and then adding structures to coordinate their work at the program and portfolio level.

Gary Gruver, Director of Engineering for FutureSmart, contrasts this approach of "trying to enable the efficiencies of small agile teams in an enterprise" with the FutureSmart team's approach of "trying to make an enterprise agile using the basic agile principles."[19] In the FutureSmart approach, while the teams ran within tight guide rails in terms of engineering practices (which we discuss in more detail in Chapter 8), there was relatively little attention paid to whether they had, for example, implemented Scrum at the team level. Instead, teams have relative autonomy to choose and evolve their own processes, provided they are able to meet the program-level target conditions for each iteration.

This required that engineering management had the freedom to set their own program-level objectives. That is, they didn't have to get budget approval to pay for process improvement work such as test automation or building out the toolchain for continuous integration. Indeed, the business wasn't even consulted on this work. All business demand was also managed at the program level. Notably, product marketing requests always went through the program-level process, without feeding work directly to teams.

Another important consideration is the way enterprises treat metrics. In a control culture, metrics and targets are often set centrally and never updated in response to the changes in behavior they produce. Generative organizations don't manage by metrics and targets. Instead, the FutureSmart management "use[s] the metrics to understand where to have conversations about what is not getting done."[20] This is part of the strategy of "Management by Wandering Around" pioneered by HP founders Bill Hewlett and Dave Packard.[21] Once we discover a problem, we ask the team or person having a hard time what we can do to help. We have discovered an opportunity to improve. If people are punished for failing to meet targets or metrics, one of the fallouts is that they start manipulating work and information to look like they are meeting the targets. As FutureSmart's experience shows, having good real-time metrics is a better approach than relying on scrums, or scrums of scrums, or Project Management Office reporting meetings to discover what is going on.

Conclusion

The Improvement Kata provides a way to align teams and, more generally, organizations by taking goals and breaking them down into small, incremental outcomes (target conditions) that get us closer to our goal. The Improvement

19 [gruver], Chapter 15.

20 [gruver], p. 38.

21 Perhaps it's better characterized as "Management by Wandering Around and Asking Questions." In the Toyota Production System, this is known as a *gemba* walk.

Kata is not just a meta-methodology for continuous improvement at the enterprise and program level; it is a way to push ownership for achieving those outcomes to the edges of the organization, following the Principle of Mission. As we show in Chapter 9, it can also be used to run large programs of work.

The key characteristics of the Improvement Kata are its iterativeness and the ability to drive an experimental approach to achieve the desired target conditions, which makes it suitable for working in conditions of uncertainty. The Improvement Kata is also an effective way to develop the capabilities of people throughout the enterprise so they can self-organize in response to changing conditions.

The FutureSmart case study shows how a large, distributed team applied the Improvement Kata meta-method to increase productivity eightfold, improving quality and substantially reducing costs. The processes and tools the team used to achieve this transformation changed and evolved substantially over the course of the project. This is characteristic of a truly agile organization.

Implementing an enterprise-level continuous improvement process is a prerequisite for any ongoing large-scale transformation effort (such as adopting an agile approach to software delivery) at scale. True continuous improvement never ends because, as our organization and environment evolve, we find that what works for us today will not be effective when conditions change. High-performance organizations are constantly evolving to adapt to their environment, and they do so in an organic way, not through command and control.

Questions for readers:

- Do you know how much time your engineering organization is spending on no-value-add activities and servicing failure demand versus serving value demand, and what the major sources of waste are?
- Must engineering teams get permission to invest in work that reduces waste and no-value-add activity across the value stream as a whole, such as build, test, and deployment automation and refactoring? Are such requests denied for reasons such as "there is no budget" or "we don't have time"?
- Does everyone within the organization know the short- and long-term outcomes they are trying to achieve? Who decides these outcomes? How are they set, communicated, reviewed, and updated?
- Do teams in your organization regularly reflect on the processes they use and find ways to experiment to improve them? What feedback loops are in place to find out which ideas worked and which didn't? How long does it take to get this feedback?

PART IV

TRANSFORM

> Everyone thinks of changing the world, but no one thinks of changing himself.
>
> — Leo Tolstoy

If you've made it from the beginning of this book to here, you should have a pretty good idea of how to apply lean concepts and principles to make great software products, and of the importance of strategy and culture in enabling the discovery and exploitation of new businesses. But to reap the maximum reward for all our efforts, lean principles and concepts need to be scaled throughout the entire organization. Only when this happens will we realize the full value of the work we have invested in, exploring new ideas and exploiting those that deliver value to customers.

We readily grasp that these concepts work well to address the needs of rapidly changing environments and fierce competition. However, it is hard to extend lean concepts to process improvement, COTS applications, and the evolution and support of internal systems, particularly systems of record. Supplier and vendor relationships present a further obstacle. The nature of our relationship with suppliers of proprietary, specialized, or customized solutions often inhibits collaboration, fast feedback, or small incremental change. We need to seek suppliers who are willing to treat us as we expect to treat our own customers. We must encourage suppliers to listen to us, understand what we need, and

experiment. They have to be willing to go on the journey of improvement with us.

To add further complexity to this problem, many of our traditional approaches to governance, risk, and compliance (GRC), financial management, procurement, vendor/supplier management, and human resources (recruiting, promotion, compensation) create additional waste and bottlenecks. These can only be eliminated when the entire organization embraces lean concepts and *everyone* works together in the same direction.

Making an enterprise lean is not a one-person or one-department show. It won't work through a special tactical task force. We can't mandate that from now on, everyone will work this way and expect them to adjust as per our implementation plan. Real lean transformation is the result of committed, fearless leaders who encourage and enable lean thinking to propagate throughout the entire fabric of the organization—not just customer-facing products. Those at the top need to walk the talk and be role models for everyone. They need to set aside egos, listen and respect contrary opinions, and build trusting relationships at all levels of the organization. This is essential for new leaders to emerge and for lean concepts and practices to become woven into the organization's culture.

People must feel empowered to make decisions that involve risk and try out new ideas, while recognizing their responsibilities to customers and maintaining alignment with the overall organization strategy. As leaders, we need to set limitations and context for everyone, but ensure they are not unduly restrictive. When everyone is united in pursuit of a common purpose, and we have empathy with our customers and put serving their needs first, most people can figure out what risks are acceptable and what are not.

Conflict arises when our espoused values do not match up with actual practice. This is where modeling the behavior we hope to see in everyone is most important. There are no formulas, instructions, or rituals that will work for everyone. Each of us needs to take time daily, maybe even several times a day, to reflect on our own actions and decide if they support our stated values and work to move us in the right direction.

A lean mindset cannot thrive in an organization with a centralized, command-and-control management style. Nevertheless, we still need to maintain visibility and transparency into what everyone is doing. It is not easy for large organizations to find this balance, and we must recognize that constant adjustment will be required. Many people within our organization will perceive this cultural shift as threatening and will push back on it. Command and control is easy; I follow the rules and if it doesn't work, it is not my fault. However, if you ask me to make decisions and take responsibility for them, someone might hold me

accountable for the mistakes I am very likely to make. Give me command and control!

If we are successful at creating a lean enterprise, we will have failures and setbacks, at all levels and by all people. If we don't, it means we haven't created a high-trust, high-performance culture and are continuing to judge our performance by vanity metrics, not real outcomes. We'll never have the culture of a learning organization if we can't be allowed to make mistakes and get worse at something before we get better.

In this part, we concentrate our discussion on how to pursue the never-ending work of transforming the enterprise. We will address some of the most common areas where we see a mismatch between lean concepts and the prevailing leadership and management principles, practices, and processes. These gaps are revealed when we face obstacles that prevent us doing better at delivering value to customers, or when we are robbed of satisfaction and fulfilment in our daily work. Our hope is that readers will be inspired to find ways to overcome these obstacles, and to share their successes—and their failures—with others.

CHAPTER 11

Grow an Innovation Culture

> The ability of your company to be competitive and survive lies not so much in solutions themselves, but in the capability of the people in your organization to understand a situation and develop solutions.
>
> — Mike Rother

> We now accept the fact that learning is a lifelong process of keeping abreast of change. And the most pressing task is to teach people how to learn.
>
> — Peter Drucker

> You know, I'm all for progress. It's change I object to.
>
> — Mark Twain

Culture is the most critical factor in an organization's ability to adapt to its changing environment. However, being intangible, it is hard to analyze and even harder to change. Every organization has its own unique culture, and there are "as many successful cultures as there are successful companies."[1] In Chapter 1 we presented the characteristics of a high-performance, generative culture. In this chapter, we discuss how to understand your organization's culture and what you can do to change it.

Culture is constantly changing in every organization. New employees and leaders join, people quit, strategies and products evolve and die, and the market

1 *The Economist*, 410, no. 8869, p. 72.

constantly shifts. The most important question is: can we mindfully evolve our organizational culture in response to these changes in environment?

To understand how to influence organization culture, we need to understand its foundations. We introduce a model of organizational culture and discuss how to measure it. We follow with strategies to kick-start organizational change, with the goal of making these strategies self-sustaining. Finally, we examine the relationship between individuals and organizations, and discuss how to hire and retain "good" people.

Model and Measure Your Culture

> CEOs can talk and blab all day about culture, but the employees know who the jerks are.
>
> — Jack Welch

In *The Corporate Culture Survival Guide*, Schein defines culture as "a pattern of shared tacit assumptions that was learned by a group as it solved its problems of external adaptation and internal integration, that has worked well enough to be considered valid and, therefore, to be taught to new members as the correct way to perceive, think, and feel in relation to those problems."[2] The "tacit" part of this definition is important—and it is what makes culture so intangible. Shanley Kane, author of *Your Startup Is Broken: Inside the Toxic Heart of Tech Culture*, provides another perspective, commenting that "our true culture is made primarily of the things no one will say…Culture is about power dynamics, unspoken priorities and beliefs, mythologies, conflicts, enforcement of social norms, creation of in/out groups and distribution of wealth and control inside companies."[3]

Even though culture is intangible, it is measureable, and there is a large body of work dedicated to precisely this task. Of course every methodology is based on an underlying model, and all models are limited to a different extent. Nevertheless, such measurements are important as a way of making culture visible and encouraging people to pay attention to it. Here are examples of work that has been done to measure culture:

- Karen E. Watkins and Victoria J. Marsick developed the *Dimensions of the Learning Organization Questionnaire* (DLOQ), which has been extensively studied in the academic literature. You can take the questionnaire for free at *http://www.partnersforlearning.com/instructions.html*.

2 [schein]

3 [kane], "Five Tools for Analyzing Dysfunction in Engineering Culture" and "Values Towards Ethical and Radical Management."

- Gallup's Q12 survey asks what they believe are "the only 12 questions that matter" to measure employee engagement. You can find the questions, along with more information, at *http://q12.gallup.com/*.
- In Chapter 1, we discussed how the *2014 State of DevOps Report* measured both job satisfaction and culture (using the Westrum model) and their impact on organizational performance. Analysis showed that Westrum's model predicted both job satisfaction and organizational performance in the context of knowledge work. Read more at *http://bit.ly/1v71SJL*.

TIP

Practicalities of Running Cultural Surveys

Whether you use a service or come up with your own survey, be careful about how much information is collected. To obtain honest answers, don't ask people to disclose identifying information. Present results only in aggregate. It may be useful to capture some demographic information so you can see, for example, how results vary between genders or roles, but only when you have numbers large enough to provide anonymity. Be mindful of how the information can work against the respondents. At one large enterprise, managers reacted to poor survey results in their department by ordering their own reports to paint them in a better light next time.

Disassociate culture surveys from pay and performance reviews. Make the aggregated results available to all employees and ensure executives set up meetings to discuss the findings and plan next steps. Run surveys annually or semiannually to provide a baseline for comparison and measurement of change over time.

Measuring organizational culture and making problems visible is the first step. Next, we must investigate *why* a culture is the way it is. For this inquiry, it is helpful to use Schein's model, which divides culture into three layers: artifacts, espoused values, and underlying assumptions (Figure 11-1).

Figure 11-1. *Layers of organizational culture*

Inconsistencies between espoused values and observed behaviors within an organization are common. Observed behaviors are better indicators of real values. Who gets rewarded for what behavior? Who gets hired, promoted, or fired? In order to understand the nature and source of the real values, we have to descend to the level of underlying assumptions. This level is hard to unpack, but it is the most important to understand.

Schein presents an exhaustive typology of tacit assumptions, of which the most important are the beliefs leaders and managers hold about workers. In his management classic *The Human Side of Enterprise*, Douglas McGregor describes two contrasting sets of beliefs held by managers he observed, which he calls Theory X and Theory Y. Managers who hold Theory X assumptions believe that people are inherently lazy and unambitious and value job security more than responsibility; extrinsic (carrot-and-stick) motivation techniques are

the most effective to deal with workers.[4] In contrast, Theory Y managers believe "that employees could and would link their own goals to those of the organization, would delegate more, function more as teachers and coaches, and help employees develop incentives and controls that they themselves would monitor."[5]

As we saw in Chapter 1, while extrinsic motivators such as bonuses are effective in a Taylorist world of routine, mechanical work, they actually *reduce* performance in the context of knowledge work. People involved in nonroutine work are motivated by intrinsic factors summarized by Dan Pink as "1. *Autonomy*—the desire to direct our own lives. 2. *Mastery*—the urge to get better and better at something that matters. 3. *Purpose*—the yearning to do what we do in the service of something larger than ourselves."[6]

What is especially problematic is that, by generating behavioral responses that align with their management style, both types of managers believe their style works best. People whose management strategy is consistent with Theory X end up with employees who are passive, resistant to change, unwilling to accept responsibility, and make "unreasonable demands for economic benefits."[7] This is a rational response by employees to not having their higher needs satisfied through work. Work becomes something to be endured in order to get a paycheck.

In an organization whose leaders share Theory Y assumptions, their job is "the creation of conditions such that the members of the organization can achieve their own goals *best* by directing their efforts towards the success of the enterprise,"[8] delivering value to customers and the organization while growing their own capabilities. Until leaders and managers with Theory X attitudes work to adopt a Theory Y mindset and demonstrate it consistently over time through their actions, they will not be able to achieve a perceptible difference in people's behavior. The story of NUMMI in Chapter 1 is a good example of this shift in mindset and behavior.

Culture is hard to change *by design*. As Schein says, "Culture is so stable and difficult to change because it represents the accumulated learning of a group—

---

4 A sophisticated and entertaining exploration of a Theory X organization and its lifecycle, based on the work of Ricky Gervais, can be found at *http://bit.ly/1v71WJq*.

5 [schein], p. 64.

6 *http://www.danpink.com/drive-the-summaries*. Pink also references a number of studies which demonstrate conclusively that extrinsic motivation reduces performance in knowledge work.

7 [mcgregor], p. 42.

8 [mcgregor], p. 49.

the ways of thinking, feeling, and perceiving the world that have made the group successful."[9]

Change Your Culture

In his revolutionary work *Pedagogy of the Oppressed*, published in 1970, Paulo Freire describes what is still the dominant model of teaching today. In this model, students are viewed as empty "bank accounts" to be filled with knowledge by teachers—not as participants who have a say in what and how they learn. This model is not designed to enable students to learn—especially not to learn to think for themselves—but rather to control the learning process, students' access to information, and their ability to critically analyze it. In this way, the education system perpetuates existing social structures and power hierarchies.

Similarly, most companies seem to treat their employees as filled-up bank accounts to be drained of skills and knowledge in service of the company's goals. This is the implication when we speak of employees as "resources" and wonder how to increase their utilization and productivity with little regard for their personal development. This kind of behavior indicates an environment in which employees exist primarily as providers of labor, not as active participants in creating value.[10] In contrast, high-performance organizations are effective at both developing and harnessing the unique capabilities of their people.

Organizations with a "bank account" attitude to employees tend to treat change in a transactional way. This all too common and flawed approach involves funding a change program which is expected to "fix" the organization so it is fit for purpose. Organizational change is treated as a product—sold by consultants, paid for by leadership, and consumed by the rest of the organization as directed.

These change programs commonly focus on reorganizing teams and reporting structures, sending employees on short training courses, and rolling out tools and methodologies across the organization. These strategies usually don't work because they are ineffective at changing people's patterns of behavior. As Mike Rother points out in *Toyota Kata*, "what is decisive is not the form of the organization, but how people act and react."[11] This is determined primarily by the actions of leadership and management. To pick some examples: are people given the autonomy to act and trusted to take risks? Is failure punished or does

9 [schein], pp. 27–28.

10 The key concept here is the idea that workers are fungible—that is, essentially interchangeable—resources. Any time you hear people referred to as "resources," this is what is implied.

11 [rother-2010], p. 236.

it lead to enquiry and improvements of our systems? Is cross-functional communication rewarded or discouraged?

We began this book by discussing the case of NUMMI, in which a broken organization was reformed under a new leadership and management paradigm. Despite rehiring the same people, NUMMI achieved extraordinary levels of quality and productivity and reduced costs. In an article for *MIT Sloan Management Review*, John Shook, Toyota City's first US employee, reflected on how that cultural change was achieved:[12]

> What my NUMMI experience taught me that was so powerful was that the way to change culture is not to first change how people think, but instead to start by changing how people behave—what they do. Those of us trying to change our organizations' culture need to define the things we want to do, the ways we want to behave and want each other to behave, to provide training and then to do what is necessary to reinforce those behaviors. The culture will change as a result… What changed the culture at NUMMI wasn't an abstract notion of "employee involvement" or "a learning organization" or even "culture" at all. What changed the culture was giving employees the means by which they could successfully do their jobs. It was communicating clearly to employees what their jobs were and providing the training and tools to enable them to perform those jobs successfully.

Shook offers his own interpretation of Schein's model, showing how people normally approach cultural change in contrast to the approach taken at NUMMI, in Figure 11-2.

NUMMI had an advantage in achieving their cultural change. The entire workforce was newly hired—with many workers having been freshly fired from their jobs at Fremont Assembly. It's hard to achieve sustained, systemic change without any crisis. In *The Corporate Culture Survival Guide*, Schein asks if crisis is a necessary condition of successful transformations; his answer is, "Because humans avoid unpredictability and uncertainty, hence create cultures, the basic argument for adult learning is that indeed we do need some new stimulus to upset the equilibrium. The best way to think about such a stimulus is as *disconfirmation*: something is perceived or felt that is not expected and that upsets some of our beliefs or assumptions…disconfirmation creates *survival anxiety*—that something bad will happen if we don't change—or *guilt*—we realize that we are not achieving our own ideals or goals."[13]

---

12 *http://bit.ly/1v720ZH*

13 [schein], p. 106.

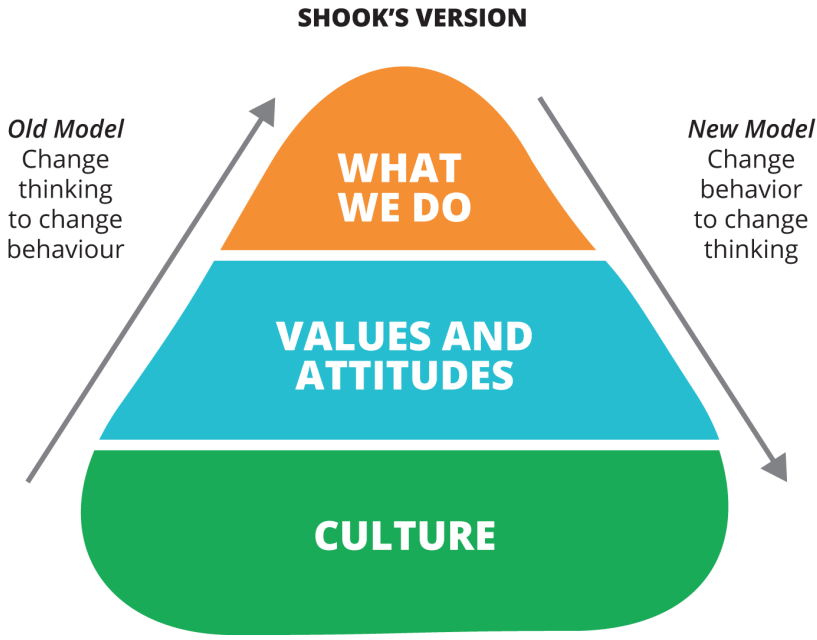

Figure 11-2. *Old and new approaches to cultural change (2010 from MIT Sloan Management Review/Massachusetts Institute of Technology, all rights reserved, distributed by Tribune Content Agency, LLC)*

Disconfirmation can come naturally from a number of sources that may threaten our survival: economic, political, technological, legal, moral, or simply a realization that we are not achieving our purpose. A common cause of unplanned disconfirmation is leaders acting in a way that contradicts their stated values. It is also possible to create disconfirmation in a controlled way through joint ventures, planned leadership activity, or by creating an artificial crisis.

Once people accept the need for change, they are confronted with the fear that they may fail at learning the new skills and behavior required of them, or that they may lose status or some significant part of their identity—a phenomenon Schein calls *learning anxiety*.

Schein postulates that for change to succeed, survival anxiety must be greater than learning anxiety, and to achieve this, "learning anxiety must be reduced rather than increasing survival anxiety."[14] Many leaders and managers make

14 [schein], p. 114.

the mistake of trying to achieve change by increasing survival anxiety. This creates an environment of fear which, in turn, results in significant amounts of energy spent on diverting blame, avoiding responsibility, or playing political games.

The most powerful systematic tool to reduce survival anxiety that we have encountered is the Improvement Kata, described in Chapter 6. It is designed for people to safely learn new skills and experiment with new ideas in the pursuit of clearly defined, measurable organizational goals. Essential to creating a high-performance culture is an environment in which mistakes are accepted as learning opportunities to build systems and processes that reduce the impact of future mistakes.

Make It Safe to Fail

Your organization's attitude to failure—whether of a change effort or simply a decision—is critical in creating an adaptive, resilient organization. Organizational theorist Professor Russell L. Ackoff noted, "It's our treatment of error that leads to a stability which prevents significant change." If people are told that making mistakes is bad, and if people are punished for them, the inevitable outcome is that they will avoid taking *any* risky decisions.[15]

In a complex, adaptive system such as an enterprise, nobody has perfect information. Every decision will have unintended consequences whose causes may be clear looking back, but are almost impossible to predict looking forward. Whenever it appears that one person is responsible for a given outcome, we should be honest and ask ourselves, "If I had been in the same situation, is it possible I would have made the same decisions?" Usually, the answer is "yes."

Rather than punishing mistakes, we must ensure that people have the necessary information to make effective decisions, find ways to limit the possible negative outcomes of decisions, and be disciplined about learning from mistakes. For example, how do managers and leaders in your organization respond to failures? Do they lead to scapegoating, justice, or enquiry?

One practice often used by organizations with high-performance cultures is a blameless postmortem run after every incident or accident. The goal of the postmortem is to improve the system so that, in similar situations in the future, people have better information and tools at their disposal and the negative impact is limited.

At the beginning of every postmortem, every participant should read aloud the following words, known as the *Retrospective Prime Directive*: "Regardless of

15 *http://youtu.be/MzS5V5-0VsA?t=6m*

what we discover, we understand and truly believe that everyone did the best job they could, given what they knew at the time, their skills and abilities, the resources available, and the situation at hand."[16] A postmortem should aim to provide:[17]

- A description and explanation of how the incident happened, from the perspective of those involved and affected, including a timeline of events and a list of contributing factors
- Artifacts (recommendations, remediations, checklists, runbook updates, etc.) for better prevention, detection, and response to improve the handling of similar events in the future

Postmortems should not attempt to identify a single root cause. The idea that a single event can be identified as the cause of a failure is a misunderstanding of the nature of complex adaptive systems. As safety experts Sidney Dekker, Erik Hollnagel, David Woods, and Richard Cook point out:[18]

> Our understanding of how accidents happen has undergone a dramatic development over the last century. Accidents were initially viewed as the conclusion of a sequence of events (which involved "human errors" as causes or contributors). This is now being increasingly replaced by a systemic view in which accidents emerge from the complexity of people's activities in an organizational and technical context. These activities are typically focused on preventing accidents, but also involve other goals (throughput, production, efficiency, cost control) which means that goal conflicts can arise, always under the pressure of limited resources (e.g., time, money, expertise). Accidents emerge from a confluence of conditions and occurrences that are usually associated with the pursuit of success, but in this combination—each necessary but only jointly sufficient—able to trigger failure instead.

Every failure is the result of multiple things going wrong—often *invisibly* (Dekker refers to complex adaptive systems "drifting into failure").[19] Every postmortem should result in multiple ideas for incremental improvement. We must also schedule a follow-up to test whether these improvements were effective,

---

16 [kerth]

17 These two points are John Allspaw's: *http://bit.ly/1e9idko*. And if you're interested in how Knight Capital lost $460m in 30 minutes, this post is worth reading in full.

18 [dekker], p. 6.

19 You can find a short guide to failure in complex systems at *http://bit.ly/1F7O3Mg*.

ideally by running an exercise simulating a similar failure, as we describe in Chapter 14.

There Is No Talent Shortage

In the tech industry it's common to hear about a "talent shortage" and the difficulty of finding "good people."[20] In this section we'll dismantle the assumptions behind these kinds of remarks. We will examine what we mean by "good people" by looking at one particular role—software engineers—and then progress to the general case.

It's a widely held belief that there is an order-of-magnitude difference between the best and the worst engineers.[21] In reality, the 10x figure is (to put it mildly) "poorly supported by empirical evidence."[22] However, once you get to the bottom of the debate over the claim, it is really about the validity, or usefulness, of individual productivity measurements in the context of an organization.

Individual productivity is most commonly measured by throughput—the time it takes to complete a standardized task under controlled conditions. This approach is premised upon a Taylorist view of work where managers define the tasks to be done and workers try to complete these tasks as rapidly as possible. Thus, old-school metrics such as lines of code per day and number of hours worked are used to measure individual productivity of software engineers. The flaws in these measures are obvious if we consider the ideal outcomes: the fewest lines of code possible in order to solve a problem, and the creation of simplified, common processes and customer interactions that reduce complexity in IT systems. Our most productive people are those that find ingenious ways to avoid writing any code at all.

In many organizations, worrying unduly about variations between individuals is futile. If there's one thing we should learn from the NUMMI case study in Chapter 1, it's that organizational culture and leadership dwarf differences between individuals. As journalist and author Malcolm Gladwell writes, "The talent myth assumes that people make organizations smart. More often than not, it's the other way around…Our lives are so obviously enriched by individual brilliance. Groups don't write great novels, and a committee didn't come up with the theory of relativity. But companies work by different rules. They

20 The title of this section is taken from a presentation by Andrew Shafer: *https://www.youtube.com/watch?v=P_sWGl7MzhU*.

21 The original reference is from [sackman], and a robust discussion and defense of the claim can be found at *http://bit.ly/1v72hvu*.

22 [bossavit], in Chapters 5 and 6, does an effective job at demolishing the existing studies and data.

don't just create; they execute and compete and coordinate the efforts of many different people, and the organizations that are most successful at that task are the ones where the system is the star." As W. Edwards Deming noted, "A bad system will beat a good person every single time."

The rate at which we can understand and solve complex problems—the key skill for which we still need people, rather than machines—is determined as much by our environment as our own skills and abilities. We can hardly blame people for failing to learn and solve problems if we limit their opportunities by organizational silos that insulate workers from each other and from customers, by long cycle times that delay feedback, by focusing on completing assigned work rather than achieving customer outcomes, and by working long hours so we have no time to try out new ideas and technologies or even talk to each other!

Given that the culture of an organization has such a dominant effect on the performance of individuals, should we care at all about the particular skills and attitudes of individuals? Instead of taking a "bank account" view that focuses on people's existing capabilities, it's more important to consider their ability to *acquire* new skills—particularly in the field of technology where useful knowledge and skills change rapidly.

Carol Dweck, Professor of Psychology at Stanford, has spent years researching the psychology of learning, development, and motivation. Her research reveals there is a way to judge how good people will be at learning new skills. Dweck discovered that our ability to learn is determined by our beliefs concerning the question: is ability innate, or can it be learned? We can observe, based on people's behavior, where they fall on a continuum between two extremes:[23]

> In a fixed mindset, students believe their basic abilities, their intelligence, their talents are just fixed traits. They have a certain amount and that's that, and then their goal becomes to look smart all the time and never look dumb. In a growth mindset, students understand that their talents and abilities can be developed through effort, good teaching, and persistence. They don't necessarily think everyone's the same or anyone can be Einstein, but they believe everyone can get smarter if they work at it.

Dweck showed through a series of experiments that our mindset determines how we decide our goals, how we react to failure, what are our beliefs about effort and strategies, and what is our attitude towards the success of others (Figure 11-3). Our mindset is particularly important in terms of our attitude to

23 *http://bit.ly/1v72nmV*

failure. People with a fixed mindset fear failure as they believe it makes their innate limitations visible to others, whereas those with a growth mindset are less risk averse by seeing failure as an opportunity to learn and develop new skills.

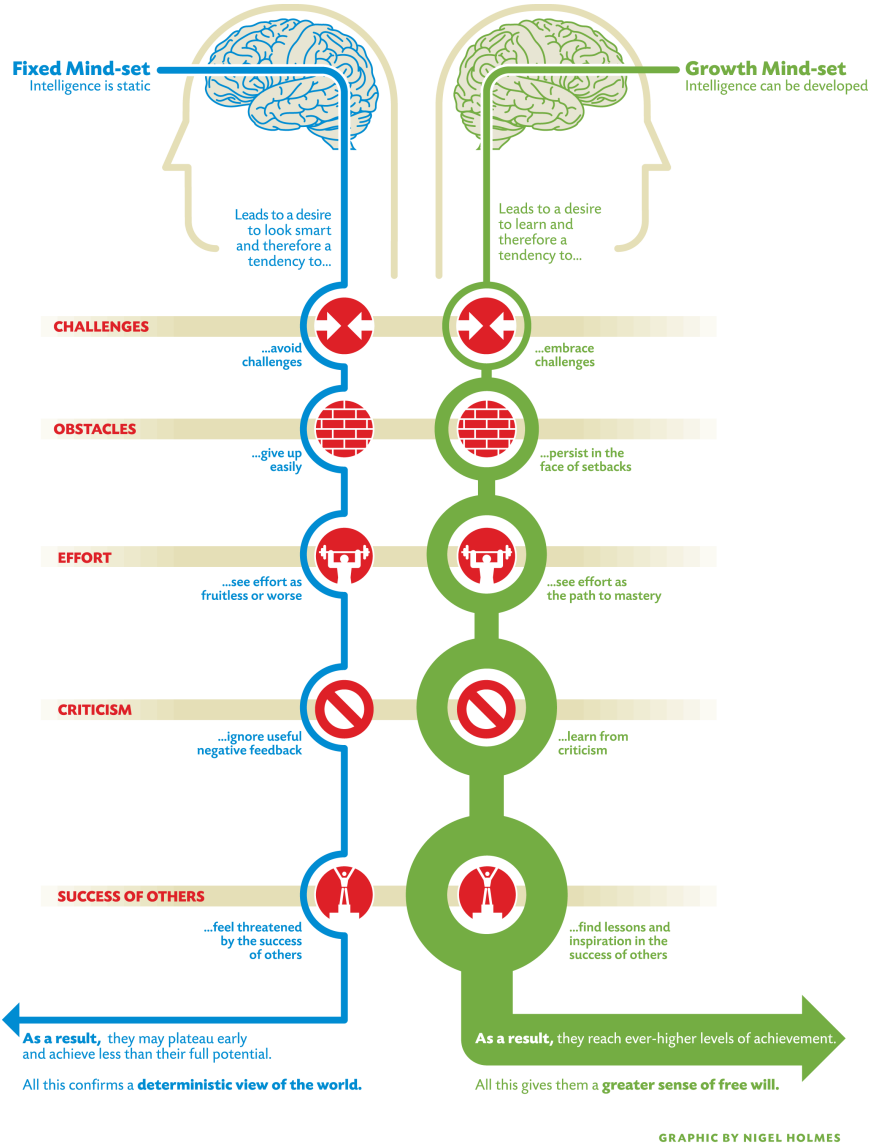

Figure 11-3. *Dweck's two mindsets, courtesy of Nigel Holmes*

The good news is that we can change our beliefs, as shown by one of Dweck's most interesting experiments.[24] Dweck's work shows that if we reward people for the effort they put into solving problems that they find challenging, it shifts them towards a growth mindset. If, in contrast, we praise and reward people for their ability to deploy their existing skills, we create a fixed mindset. This has important implications both for people managers and for HR departments, particularly in the context of performance reviews.

You can be sure that the behavior and attitudes of the people in your organization—your organization's culture—affect the mindsets of the individuals within it, and thus their ability to learn. Thus, organizational culture determines not just the productivity and the performance of the people working in it, but also their ability to gain new skills, their attitude to failure and new challenges, and their goals. Setting people stretch goals that require them to learn new skills, while providing them with support, training, and slack time to reduce learning anxiety, and creating a culture in which collaboration is rewarded and failure leads to reflection and improvement rather than blame—all this works to instill a growth mindset in employees and must be a key goal of organizational change.

Dweck's work tells us there are indeed "A-players" and "B-players." A-players are simply people with a growth mindset who, upon joining a team, will try to discover how to make the team successful, working to acquire the necessary skills in the process. In contrast, people with a fixed mindset—the true B-players—are the biggest barrier to organizational change and continuous improvement. These are the kind of people who resist experimentation saying that others' approaches "can't work here." They are also likely to hire people they perceive as worse than them so as to avoid challenges to their status and identity. While such people are capable of changing their mindset, they can also poison attempts to change culture, holding back high-performing teams. To reduce learning anxiety during change efforts, it must be widely publicized that support and resources will be available to help people acquire new skills, that no one will lose their job if they are willing to learn, and that those wishing to leave will receive a generous severance package.

Ultimately, the most important responsibility of an organization's leaders is for its culture, demonstrated by the way they treat others. For example, Dweck argues that while Steve Jobs possessed a growth mindset when it came to his own abilities, he had a fixed mindset attitude towards others: "He wanted

---

24 *http://gladwell.com/the-talent-myth* is an article well worth reading in its entirety. Dweck's research also has important implications for how we bring up our children, particularly girls who are often praised for being "good" or "pretty," creating a fixed mindset. This is just one way in which implicit biases reinforce each other. See *http://bit.ly/1zkRLOK*.

them to be perfect and they lived in fear of coming to him and getting his disapproval, instead of his approval."[25]

How Google Recruits

Dweck's work demands that we rethink recruiting. We should not hire people solely on the basis of the skills they already possess. This is particularly short-sighted in the software industry where technology, and thus the needed skills, change so rapidly. Neither should we be using brainteasers or test scores, which Laszlo Bock, senior vice president of people operations at Google, describes as "worthless as a criteria for hiring…They don't predict anything."[26] Google has done a great deal of research into what makes for an effective recruiting process in the context of technology. The top three criteria are:[27]

- *Learning ability,* including the ability to "process on the fly" and "pull together disparate bits of information."
- *Leadership,* "in particular emergent leadership as opposed to traditional leadership. Traditional leadership is, were you president of the chess club? Were you vice president of sales? How quickly did you get there? We don't care. What we care about is, when faced with a problem and you're a member of a team, do you, at the appropriate time, step in and lead. And just as critically, do you step back and stop leading, do you let someone else? Because what's critical to be an effective leader in this environment is you have to be willing to relinquish power."
- *Mindset.* "Successful bright people rarely experience failure, and so they don't learn how to learn from that failure…They, instead, commit the fundamental attribution error, which is if something good happens, it's because I'm a genius. If something bad happens, it's because someone's an idiot or I didn't get the resources or the market moved."

Bock goes on to observe that the most successful people at Google "will have a fierce position. They'll argue like hell. They'll be zealots about their point of view. But then you say, *here's a new fact,* and they'll go, *Oh, well, that changes things; you're right.*" This reflects the advice of Paul Saffo, Director of the Palo Alto Institute for the Future, who says that "to deal with an uncertain future and still move forward," people should have "strong opinions, which are weakly held."[28]

Google's recruiting strategy is liberating because it enormously expands the pool of qualified applicants. Instead of looking for "purple squirrels" with precisely the skills and experience required for a job, we should look for people who can rapidly acquire the necessary skills and then invest in an environment that enables them to do so.

25 *http://bit.ly/1v72nmV*

26 *http://nyti.ms/1v72xuz*

27 Quotations are from *http://nyti.ms/1v72sHl*.

28 *http://bit.ly/1v72zTg*

Growing Talent

The "talent shortage" problem is solved by creating an environment in which people can learn on the job, and hiring people with a growth mindset. Investing in employee development is one of the few opportunities enterprises have to create a competitive advantage over startups (the others being research and development, and the pursuit of optionality in Horizon 3, as described in Chapter 2). There are many ways in which enterprises can invest in people:

Help employees create and update personal development plans
: To help employees take control of their own development and ensure that managers know how to help them, it's essential that they, their managers, and people that give them feedback understand their career goals. Creating and regularly updating a simple personal development plan is the foundation of employee development.

Separate performance reviews from compensation reviews
: The goal of performance reviews is to provide an opportunity for employees to get feedback on their progress towards their personal development goals, update their goals, and discuss them with their line manager. Coupling performance reviews with compensation reviews, and particularly the practice of "stack-ranking" employees, is based on outmoded ideas of extrinsic motivation which encourage employees to compete rather than cooperate with each other, and reduce employee engagement.

Facilitate regular feedback
: Employees should share informal feedback on a regular basis to help each other move towards their personal goals. Good feedback is timely, designed for the benefit of the receiver, and given with permission. In a formal process (such as during a performance review, official reprimand, or exit interview), nobody should hear feedback that they have not already received informally.

Give employees access to training funds
: Employees learn through different channels and should have easy access to funds that enable them to buy books, attend conferences and training, or engage in other activities that help them move towards their personal development goals. The conditions for spending should be as liberal as possible, within the limitations of applicable tax regulations.

Give employees time to pursue their own goals
: Many innovative organizations reserve time for people to work on whatever they want. 3M has allowed employees to spend 15% of their time on their own projects since 1948. The Post-It Note is just one of the

innovations created as a result of this initiative.[29] In *2004 Founders' IPO Letter*, Google's Sergey Brin and Larry Page write, "We encourage our employees, in addition to their regular projects, to spend 20% of their time working on what they think will most benefit Google. This empowers them to be more creative and innovative. Many of our significant advances have happened in this manner. For example, AdSense for content and Google News were both prototyped in '20% time.' Most risky projects fizzle, often teaching us something. Others succeed and become attractive businesses."[30]

Norman Bodek tells a story about Taiichi Ohno closing down a warehouse at a Toyota subsidiary: "Get rid of this warehouse and in one year I will come back and look! I want to see this warehouse made into a machine shop and I want to see everyone trained as machinists."[31] Bodek reports that Ohno's orders were carried out, and within one year the warehouse had been replaced with a machine shop and the workers retrained. In line with the standard post-World War II Japanese corporate policy of providing people with jobs for life, Toyota expects to retrain people to do different types of work throughout their careers. Employees at Toyota understand that part of their job is to learn new skills. Toyota provides the necessary training and support for this, removing a great deal of the learning anxiety that is the most serious barrier to creating a learning organization and organizational change. Most importantly, when people are treated with respect and are given opportunities to pursue autonomy, mastery, and purpose, they become highly motivated to deliver value. Employee job satisfaction is the best predictor of organizational performance.

Eliminate Hidden Bias

Another major contributor to the "talent shortage" in technology is the large number of qualified people who decide not to enter the field or quit prematurely. Look at your technology teams and notice that women, in particular, are strongly underrepresented, as are non-white people in the US and the EU. Given that "biological sex differences in inherent aptitude for math and science are small or nonexistent,"[32] and the same holds true for differences between races, what is the cause of this underrepresentation?

A number of studies done on recruitment processes aiming to hire on merit universally show that our implicit gender bias plays a strong role in rejecting

29 *http://solutions.3m.com/innovation/en_US/stories/time-to-think*

30 *http://investor.google.com/corporate/2004/ipo-founders-letter.html*

31 [bodek], p. 29.

32 See [ceci] for the most recent of a number of studies cited in [moss-racusin].

suitably qualified women. In a study performed in 2012, researchers took 127 biology, chemistry, and physics professors from across the USA and gave them job application materials for an undergraduate science student applying for a job as a science laboratory manager. The materials were all identical, but were randomly assigned either a male or female name. Participants were asked to rate the student's "competence and hire-ability, as well as the amount of salary and the amount of mentoring they would offer the student."[33] The results are reproduced in Figure 11-4. Perhaps most interestingly, both male *and* female professors demonstrated the same bias, showing that it is not intentional or explicit but rather "shaped by implicit or unintended biases, stemming from repeated exposure to pervasive cultural stereotypes that portray women as less competent." Other studies have shown the same effects in different domains, as well as a similar effect with regard to race.[34]

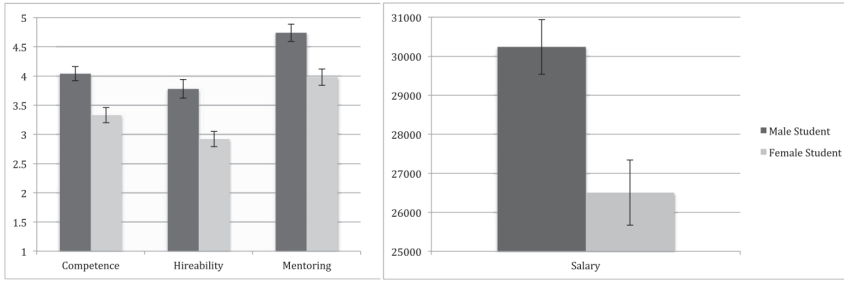

Figure 11-4. *The effects of implicit gender bias on hiring*

These implicit biases aren't limited to recruitment or gender. Implicit bias and unequal access to resources act at every stage of our educational[35] and professional lives, resulting in white male domination in science, technology, engineering, and mathematics (STEM) fields.[36] A representative survey of 19,000 people carried out in the USA by the Level Playing Field Institute between 2001 and 2006 found that the annual cost to US businesses attributable to voluntary turnover of managers and professionals due solely to unfairness was $64 billion. Respondents cited the following behaviors: rudeness, having

33 [moss-racusin]

34 See, for example, [bertrand] and *http://www.eurofound.europa.eu/ewco/2008/02/FR0802019I.htm* which references [cediey] on implicit race bias and *http://bit.ly/1v72MG7* which references [goldin] on the effect of blind auditions in increasing the number of women in orchestras.

35 University of California, Berkeley, recently redesigned its introductory computer science course, leading to the enrollment of 106 women and 104 men: *http://bit.ly/1v72O0L*.

36 In general, highly compensated professions tend to be male dominated.

coworkers at a similar or higher level who are less educated or less experienced, others taking credit for your work, being given assignments that are usually considered below your job level, feeling excluded from the team, and being stereotyped.[37]

What can we do about this? Here is a selection of strategies that have proven useful.[38] For further reading, consult Freada Kapor Klein's *Giving Notice: Why the Best and Brightest are Leaving the Workplace and How You Can Help them Stay*:

Ensure equitable pay

It's impossible to make exact comparisons between individuals, so instead examine salaries by role. Compare the average salary for white men within a particular role (such as UX analyst or senior engineer) with the average salary of people from underrepresented groups. Correct any discrepancies you find. Netflix' annual compensation review process follows a simple rule: every employee's salary is adjusted to "top of market" by ensuring they are paid "more than [any other company] likely would; as much as a replacement would cost; as much as we would pay to keep them if they had a higher offer for elsewhere."[39] If implemented comprehensively, this practice has the effect of redressing pay inequality for historically disadvantaged groups.

Create target conditions for recruitment and promotion

The Improvement Kata can and should be used as part of efforts to increase diversity. Target conditions for hiring and promotion of underrepresented groups are one example of an appropriate use of this tool. One large enterprise wanted to improve the number of women in senior management positions. To avoid accusations of positive discrimination, they didn't create a quota for the positions, but they did impose a target condition for the proportion of women on the list of candidates (for example, "50% of the candidates for the post must be women").[40] A similar approach can be used for recruiting and team mix.

Monitor tenure, rate of advancement, and job satisfaction

Gather data on the average tenure of white men compared to people from underrepresented groups. Look to see how long it takes different groups to receive promotions. Find out what proportion of each underrepresented

37 [klein], pp. 7–8.

38 An excellent summary of why women leave the tech industry and what to do about it can be found at *http://bit.ly/1toep4k*.

39 *http://www.slideshare.net/reed2001/culture-1798664*

40 *http://bit.ly/1v72Rtt*

group has at least one other person reporting to them. Analyze your job satisfaction survey to reveal differences between demographics. Higher employee churn from underrepresented groups, longer time to promotion, and lower job satisfaction are clear indicators of (at best) an implicit bias within your organization.

Regularly review policies, interactions, and HR processes
: Implicit bias doesn't just play a role in recruiting—it pervades the corporate environment. To take just one example, women are much more likely to receive critical feedback in performance reviews (the word "abrasive" is almost exclusively used in feedback to women). Similar patterns are apparent in feedback to other underrepresented groups.[41] It's essential to create clear policies, have leaders publicly and regularly set expectations for acceptable behavior, and ensure they model appropriate behavior and are seen to take action in the event of inappropriate behavior. Hire an external expert to review interactions, policies, and HR processes, make recommendations, and return regularly to monitor implementation and review progress.

Conclusion

In a high-performance organization, employees enjoy and take pride in their daily work, and leaders and managers are dedicated to supporting employees in their pursuit of the organization's purpose. No organization does this perfectly, and those that do best are constantly working to get better.

To create this kind of environment, we must address the behavior of everyone in the organization, starting with executives. As John Kotter observes, "a majority of employees, perhaps 75 percent of management overall, and virtually all of the top executives, need to believe that considerable change is absolutely essential."[42] The essence of a lean mindset is understanding that this should be the case not just in a crisis but *all the time*. Change, improvement, and development are habitual in a truly lean organization.

Changing culture is achieved by the deliberate, repeated, mindful practice of everyone in the organization. Leaders and managers must facilitate this by investing in employees' development and creating conditions to support people working together to continuously improve processes, knowledge, and the value delivered to customers. Finally, it's essential that leaders model the behaviors they expect the rest of the organization to adopt. Leaders whose actions

41 See *http://bit.ly/1v72Q8Rs* and *http://bit.ly/1v72WNz*.

42 [kotter], p. 51.

contradict their words—particularly when their status is threatened or in times of stress—will lose the trust of their people.

Questions for readers:

- Does your organization send out an anonymous survey (at least annually) to measure job satisfaction and other indicators of culture? Are aggregated results published to estimate progress towards targets for job satisfaction, diversity, and real cultural change? Are the results discussed and acted upon?
- What happens when something goes wrong? Is there a systematic process to learn from accidents in order to improve the systems, or do managers focus on assigning blame?
- What does your organization do to invest in the long-term growth of employees?
- Does you company see culture change as continuous or event based? What practices could you start to move to a continuous model?
- Does your organization hire those with particular skills and experience, or people with the capability and attitude to learn the relevant skills to help their team succeed?
- Has your organization invested in reducing and eliminating the effects of systematic implicit biases? How are you measuring your progress?

CHAPTER 12

Embrace Lean Thinking for Governance, Risk, and Compliance

> All things are subject to interpretation. Whichever interpretation prevails at a given time is a function of power and not truth.
>
> — Friedrich Nietzsche

> Trust is not simply a matter of truthfulness, or even constancy. It is also a matter of amity and goodwill. We trust those who have our best interest at heart, and mistrust those who seem deaf to our concerns.
>
> — Gary Hammel

We often hear that Lean Startup principles and the techniques and practices we suggest in this book would never work in large enterprises because of governance. "This won't meet regulatory requirements." "That doesn't fit in our change management process." "Our team can't have access to servers or production." These are just a few examples of the many reasons people have given for dismissing the possibility of changing the way they work.

When we hear these objections, we recognize that people aren't really talking about governance; they are referring to processes that have been put in place to manage risk and compliance and conflating them with governance. Like any other processes within an organization, those established for managing

governance, risk, and compliance (GRC)[1] must be targets for continuous improvement to ensure they contribute to overall value.

There are many large enterprise organizations that have been able to apply lean engineering practices and develop a culture of experimentation as we have described earlier. They are subject to the same level of regulatory compliance and review as others. So we know it can be done.

In this chapter, we aim to guide you through the maze that is GRC, particularly as it relates to managing the concepts and practices required to be a lean enterprise. This area is sometimes poorly understood by those who have not made GRC their career focus, so we present some background to help you reach a common understanding with GRC teams. With that, it should be easier to discuss how GRC processes and controls can be improved to allow product teams to continuously explore and improve their work. We provide some examples of how lean concepts and principles can be applied to improve GRC processes, resulting in better governance and reduced overall risk, while still meeting compliance.

Throughout this chapter, we refer to "GRC teams." For clarity, our discussion and examples focus on teams that strongly influence how technology can be used within organizations; the more common ones are the PMO, technical architecture, information security, risk and compliance, and internal audit teams.

Understanding Governance, Risk, and Compliance

In the introduction to Part I, we stated that the primary responsibility of leaders is to steer the larger organization towards its goals, adjusting course as necessary. This is governance. Unfortunately, within organizations the term *governance* is often misused and conflated with management theories, models, and processes designed to meet the needs of a bygone era.

Governance is about keeping our organization on course. It is the primary responsibility of the board of directors, but it applies to all people and other entities working for the organization. It requires the following concepts and principles to be applied at all levels:

Responsibility
: Each individual is responsible for the activities, tasks, and decisions they make in their day-to-day work and for how those decisions affect the overall ability to deliver value to stakeholders.

1 Typical GRC processes include access control, solution delivery (project management), change management, and related activities to reduce risks with the use of IT.

Authority or accountability
: There is an understanding of who has the power and responsibility to influence behaviors within the organization and of how it works.

Visibility
: Everyone at all times can view the outcomes achieved by the organization and its components, based on current and real data. This, in turn, can be mapped to the organization's strategic goals and objectives.

Empowerment
: The authority to act to improve the delivery of value to stakeholders is granted at the right level—to the people who will deal with the results of the decision.

Risk is the exposure we run for the possibility of something unpleasant occurring. We all manage risks daily, at work, home, and play. As it is impossible to eliminate every risk, the question to be answered in managing risk is, "Which risks are you willing to live with?" As you take steps to mitigate risk in one area, you inevitably introduce more risk in another area. A classic example of this is restricting development team access to hardware and forcing them to rely on a separate centralized infrastructure team to set up access and environments for testing or experiments. This may be effective for the server support team's goal of reducing the risk of instability within systems, but it increases the risk of delayed delivery as teams have to submit requests to other teams and wait for them to be fulfilled.

Compliance is obedience to laws, industry regulations, legally binding contracts, and even cultural norms. The intention of mandated compliance is usually to protect the interest of stakeholders with regard to privacy of information, physical safety, and financial investments. When bound by law, regulation, or contract, compliance is not optional. If we choose not to comply, we increase our risk of fines, operational shutdowns, or damage to our reputation. In extreme cases, jail terms can be the outcome of knowingly and systematically misrepresenting an organization's compliance.

Management Is Not Governance

COBIT 5[2] clearly explains the difference between governance and management.

Governance ensures that stakeholder needs, conditions, and options are evaluated to determine balanced agreed-on enterprise objectives to be achieved; sets direction through prioritization and decision making; and monitors performance and compliance against agreed-on direction and objectives.

Management plans, builds, runs, and monitors activities in alignment with the direction set by the governance body to achieve the enterprise objectives.

For example, governance involves creating the vision and goals for implementing technology changes at a rate that will allow the business to succeed. It defines what should be measured to determine if we are headed in the right direction to achieve our goals. Management determines how the organization will achieve that vision. In the case of technology changes, that includes structuring of the delivery teams, their boundaries, and what level of decision they are empowered to exercise. Will it be a single, one-size-fits-all, top-down driven process, or will teams be granted autonomy and empowered to make decisions without having to wait for high-level approvals? Good GRC management maintains a balance between implementing enough control to prevent bad things from happening and allowing creativity and experimentation to continuously improve the value delivered to stakeholders.

Take an Evolutionary Approach to Risk Management

A struggle we often experience when implementing GRC structures and processes for compliance is thinking of them as something carved in stone, rather than something that should be changed, modified, and improved. To enable good governance, changes to GRC processes must happen over time in response to the changing needs of the organization and the market environment within which it exists.

When done well, GRC management processes improve value delivery through effective risk management. The intent is to improve communication, visibility, and understanding of who is doing what, when, how, and why, as well as the outcomes of the work that is done. This is strongly aligned with what product

2 As set out in [COBIT5], COBIT formally stands for Control Objectives for Information and Related Technology. It strives to provide an end-to-end business view of the governance of enterprise IT. Auditors as well as risk and compliance teams use the framework and related tools to create and assess governance over the use of technology in delivering value. For more information, see *http://www.isaca.org/cobit/pages*.

delivery teams are trying to achieve. The question then becomes: why are GRC processes viewed as blockers when looking for ways to improve our productivity and the value we deliver to customers and our organization?

Unfortunately, many GRC management processes within enterprises are designed and implemented within a command-and-control paradigm. They are highly centralized and are viewed as the purview of specialized GRC teams, who are not held accountable for the outcomes of the processes they mandate. The processes and controls these teams decree are often derived from popular frameworks without regard to the context in which they will be applied and without considering their impact on the entire value stream of the work they affect. They often fail to keep pace with technology changes and capabilities that would allow the desired outcomes to be achieved by more lightweight and responsive means. This forces delivery teams to complete activities adding no overall value, create bottlenecks, and increase the overall risk of failure to deliver in a timely manner.

Apply Lean Principles to GRC Processes

As with everything else we address in this book, the journey to apply lean principles to GRC processes—and the ensuing results—will look different in every organization, depending on the nature of our business and where we operate. There is no cookbook recipe that fits all circumstances (as reputable frameworks like ITIL[3] and COBIT explain). However, lean principles and concepts can be applied to any GRC management process: visualizing the value stream, increasing feedback, amplifying learning, empowering teams, reducing waste and delays, limiting work in process, making small incremental changes, and continuously improving to achieve better outcomes.

A natural tension exists between GRC teams—charged with recommending and advising on how to reduce risks and meet compliance for applicable laws and regulations—and the rest of the organization who just want to get work done, the sooner the better. Tension can be good, though. It sparks creativity, but that creativity is only good if all parties involved know and strive to meet common objectives and are ultimately measured by the same standard. When tension is bad, the result is less collaboration, visibility, and compliance as individuals and teams develop secret ways to circumvent GRC processes. This leads to decisions based on inadequate or inaccurate information, which weakens overall governance.

---

3 ITIL (Information Technology Infrastructure Library, see *http://www.itil-officialsite.com*) is a framework, evolving over 20 years, providing recommended sets of practices for managing IT based on experience from both public and private sectors. It is largely used by IT management and practitioners.

The GRC teams' goals and objectives usually result in more work for all teams. Some of this is good. Upfront attention to risks, threats, and controls can save a lot of pain during the final steps towards production. Being able to prove we have adequate control measures in place is important during audits and helps keep us in compliance. The challenge is to find the correct balance of control that allows teams to move forward quickly and keeps risks related to compliance down to an acceptable level.

Define the Value of GRC Processes from the Customer Perspective

To get value out of GRC processes such as access control, technical change management, and solution delivery lifecycle, we must always start with a shared understanding of our organization's goals, values, and the intended outcomes of the process. We need a common view of how our daily work contributes to these at the organizational level, no matter with which team we associate ourselves. This means our GRC teams need to take responsibility for the outcomes (good and bad) of compliance and risk management activities and their impact on the ability of teams to deliver in a timely manner. As well, product delivery teams need to understand the language, intent, and purpose of the processes and controls established for compliance and governance. Only then will these teams, who are usually viewed as working at cross-purposes, be able to "stop fighting stupid and make more awesome."[4]

Thus, GRC teams must view themselves as members of the product delivery team, learn about the capabilities of the technology and techniques used in lean engineering, and help teams leverage them to provide evidence of being in compliance without creating waste and bottlenecks. At the same time, the entire delivery team needs to start paying attention to the language and frameworks used by GRC teams to understand what exactly it is that the GRC teams are trying to achieve.

We have seen a lot of waste and destructive tension between GRC and delivery teams because many GRC processes and management practices are disconnected from how teams work. Typically, GRC teams focus on performing and measuring compliance (for example, by asking, "Did everyone follow the activity as described in our framework?"), not on improving the outcomes ("Are we doing what will allow us to meet compliance *and* continue to deliver value in a timely fashion?").

4 Jesse Robbins, *http://www.infoq/presentations/Hacking-Culture*

TIP

Avoid the "Wouldn't It Be Horrible If" Approach to Risk Management

In *How to Measure Anything*, Douglas Hubbard reports Peter Tippet of Cybertrust discussing "what he finds to be a predominant mode of thinking about [IT security]. He calls it the 'wouldn't it be horrible if…' approach. In this framework, IT security specialists imagine a particularly catastrophic event occurring. Regardless of its likelihood, it must be avoided at all costs. Tippet observes: 'since every area has a "wouldn't it be horrible if…" all things need to be done. There is no sense of prioritization.'"[5]

When prioritizing work across our portfolio, there must be no free pass for work mitigating "bad things" to jump to the front of the line. Instead, quantify risks by considering their impacts and probabilities using impact mapping (see Chapter 9), and then use Cost of Delay (see Chapter 7) to balance the mitigation work against other priorities. In this way we can manage security and compliance risks using an economic framework instead of fear, uncertainty, and doubt.

GRC teams are measured by "Are we compliant?"; product teams are measured by "How fast can we deliver value through use of technology?" Both of these are wrong because they measure a team's performance from an isolated functional perspective and not as the net value for the organization. It is easy to be compliant with laws when GRC teams are allowed to mandate processes and force all boxes to be ticked. However, when team performance measures are not aligned at the organizational level, we can be compliant and still make remarkably bad decisions about delivering value to stakeholders. This is truly ironic, as most related laws and regulations have been established with the intent to protect and improve value to stakeholders.

Rules-based Approaches Lead to Risk Management Theater

When GRC teams do not take a principles-based approach and instead prescribe the rules that teams must blindly follow, the familiar result is *risk management theater*: an expensive performance that is designed to give the appearance of managing risk but actually increases the chances of unintended negative consequences.

At one large European enterprise we worked at, the change approval process involved developers filling in a spreadsheet with seven tabs, which was emailed to a change manager in another country who then decided whether or not to approve it. The change could not proceed without this approval, and if the form was not filled out completely it got sent back. The change manager did not really understand the contents of the spreadsheet; before approving, he relied on conversations with the developers to determine what were the risks and whether the planned mitigation activities were appropriate. The developers knew this and did the minimum possible amount of

5 [hubbard], p. 188.

work to fill in the spreadsheet, often just changing the date and title on a previous submission and sending it back as a new request. The change manager knew the developers were doing this, but it made no difference to him so long as the documented process was followed to the letter. It added zero value in terms of risk management, while making it unnecessarily painful for the team to get their changes live. However, compliance was being met through the "evidence" documented on the change request. The real value was realized in the conversations and completing mitigation activities before the change proceeded.

When product teams push back on risk management theater, a common response is that it is required by some popular framework such as ITIL or COBIT, or by a law or regulation such as Sarbanes-Oxley. However, with a few exceptions, neither frameworks nor laws prescribe particular processes. For example, many people think that segregation of duties[6] is required by Sarbanes-Oxley section 404, so organizations set up elaborate controls over access to IT systems and environments to meet their interpretation of what this means. In fact, nowhere in the act—nor in the SEC rules that were created through the act—is segregation of duties mentioned.

If you find that you are expected to follow a process that compromises your ability to do a good job, it's worth actually reaching out to the people who created the process to discuss its intent. Return to the Principle of Mission discussed in Chapter 1 and use it as an opportunity to collaborate, build relationships, and develop a shared understanding. You may be surprised to discover that you are able to have a productive conversation about how to meet their goals in a different way, or indeed to see if your work is even in scope for the law or regulation in question. If you are told that a particular process is "required" by some regulation, politely ask where you can find more information about that requirement. In many cases, onerous rules and GRC processes that are put in place are simply somebody's *interpretation* of what is required, not mandated by the regulation in question.

Map the Value Stream, Create Flow, and Establish a Pull System

With a shared understanding of GRC processes and product delivery team goals and methods, the collaboration to achieve organization-level goals can really begin. As discussed in Chapter 7, value stream mapping is a powerful tool that can be used to provide us with a view of the current state and identify areas for improvement. In the context of GRC processes, it is important to layer these on top of the delivery team activities and understand how they influence the ability of the team to get their work done.

6 *Segregation of duties* is a concept that seeks to prevent errors and malicious activities by an individual by requiring at least two people to complete any end-to-end transaction. Another way to approach it is to ensure no one person can complete a transaction without it being detected or controlled by at least one other person.

Most GRC processes are designed in isolation to apply controls such as required approvals, limited access, segregation of duties, monitoring, and review of activity. These are meant to provide visibility and transparency into who does what, when, and with what authority. More importantly, the frameworks commonly used by GRC teams to create the processes emphasize improving overall efficiency and effectiveness for the organization. Unfortunately, many of the processes and controls do the exact opposite when considered in the larger end-to-end value chain.

The Wrong Control Interrupts Flow

Controls can be preventive in nature by the application of a barrier. Alternatively, they can be detective—monitoring and reviewing events after they occur, and eliciting an appropriate response to the discovery of potential exceptions such as errors, omissions, or malicious actions.

Many of us make the mistake of thinking that preventive controls are more effective: if we can create barriers or take away people's ability to do things, it won't happen. The reality is, people need to get things done. If you try to stop them, many will get creative and figure out ways to work around whatever barriers have been put in place. The reactive response is then to lock everything down even more, which emboldens further creative underground solutions to get the work done, fomenting a subversive culture of risky behavior. A good example is teams who will share an elevated user ID and password to access different environments. It would be far better to give each team member access under their own IDs and then monitor their use of those privileges.

An even more tragic outcome of too many preventive controls is when teams just stop caring and assume an automaton mode of operation, abandoning all efforts to make things better.

Preventive controls, when executed on the wrong level, often lead to unnecessarily high costs, forcing teams to:

- Wait for another team to complete menial tasks that can be easily automated and run when needed
- Obtain approvals from busy people who do not have a good understanding of the risks involved in the decision and thus become bottlenecks
- Create large volumes of documentation of questionable accuracy which becomes obsolete shortly after it is finished
- Push large batches of work to teams and special committees for approval and processing and then wait for responses

If preventive controls are not executed properly and consistently, they are no longer effective. They must be continuously monitored to ensure they have

been applied correctly and are still relevant. Without monitoring and resulting corrective actions, preventive controls are less effective than well-executed detective controls such as ongoing monitoring, early and frequent testing and review, and highly visible measurement of outcomes.

Although relying on preventive controls may contribute to a false sense of security, they are extremely valuable when applied at the right level, and are the best solution in certain circumstances. However, they should never be applied unilaterally but only in conjunction with other controls and to the correct level of granularity, and we must always consider their effect on the ability of teams to get their work done.

Therefore, when we perform value mapping of governance processes on top of delivery team processes, we need to look carefully at all of the controls and ask two questions:

- Is the intent of the control being met?
- Is it truly contributing to overall effectiveness and efficiency of the organization?

We need to look carefully at the level of authority granted to our teams. The goal is to bring the approval decisions to the right level and give teams as much authority as possible to enable them to keep moving. This involves defining boundaries and making sure the team knows how and when to escalate decisions that fall outside their authority. We also need to make sure documentation is kept to a sane level and, when done, make sure it is accessible, easy to understand, and updated as required, preferably automatically.

"Trust, but verify"[7] is a concept that is gaining acceptance in GRC circles. Instead of preventing teams from accessing environments and hardware so they can't do anything bad, we trust people to do the right thing and give the team access and control on the systems and hardware they need to use daily. We then verify the team is not abusing their authority by developing good monitoring and frequent review processes to ensure the established boundaries are observed and there is complete visibility and transparency built into the team's work.

7 This saying, popularized by Ronald Reagan, is originally a Russian proverb.

Reducing Feedback Loops on Compliance Activities

Meeting compliance for Information Security has been a thorn in the side of many delivery teams. In the spirit of the big bang project delivery methodology, the security team is brought in at the latest possible moment—days before we go live—to run a final code review for security vulnerabilities and required compliance.

The Information Security community now realizes this approach doesn't work. On most products, there is just too much complexity and volume to complete a meaningful review. When vulnerabilities or other breaches in compliance are discovered this way, it is generally too late to do much about it. It becomes more risky to fix the vulnerabilities in a fragile system, or wait for the changes, than it is to allow the vulnerabilities to go to production with a promise to fix them later.

To meet compliance and reduce security risks, many organizations now include information security specialists as members of cross-functional product teams. Their role is to help the team identify what are the possible security threats and what level of controls will be required to reduce them to an acceptable level. They are consulted from the beginning and are engaged in all aspects of product delivery:

- Contributing to design for privacy and security
- Developing automated security tests that can be included in the deployment pipeline
- Pairing with developers and testers to help them understand how to prevent adding common vulnerabilities to the code base
- Automating the process of testing security patches to systems

They also create their own environments for performing mandatory code reviews and security testing so they don't block the team from performing other work while this is done.

As working members of the team, information security specialists help shorten feedback loops related to security, reduce overall security risks in the solution, improve collaboration and the knowledge of information security issues in other team members, and themselves learn more about the context of the code and the delivery practices. Everybody wins.

As we become better at creating flow for teams by changing governance processes, GRC teams benefit as well. Using controls designed in collaboration with GRC teams, product delivery teams are able to embed evidence of true compliance into daily work and tools, and do away with risk management theater. As we do with functional and performance quality, we build evidence of compliance into our daily work so we don't have to resort to large batch inspections after most of the work has been done.

The net effect for GRC teams is that they can now pull information related to compliance from product delivery teams at any time without interrupting the team's overall workflow, unless something untoward or unaccountable seems to be happening. Annual audits are less painful because the delivery teams understand the intent of the controls the auditors are asking for and can give evidence of meeting that intent through their processes.

By using an economic framework (such as Cost of Delay, discussed in Chapter 7) we can quantify the economic trade-offs we make when we implement controls to mitigate risk. This allows us to prioritize GRC work against the other kinds of work we do—and thus pull additional work required for compliance at the right time for the business.

Case Study: PCI-DSS Implementation at Etsy

Etsy is an online handmade and vintage marketplace with over $1bn in gross merchandise sales in 2013. In Etsy's high-trust culture, developers normally push their own changes live—indeed, as part of onboarding new engineers, developers use the automated deployment system to update their profile on the live site within their first few days. Engineers are also allowed to work on—and have access to—all parts of the system.

However, since Etsy processes credit-card transactions, it is subject to PCI-DSS, an industry standard that is quite prescriptive in how to manage systems that store or transmit payment cardholder data (these systems are known as the cardholder data environment, or CDE). For example, the CDE *must* be physically segregated, and there *must* be segregation of duties for people who work on systems within the CDE.

Segregation of duties is usually interpreted to mean (among other things) that developers should not have access to the production database and should not be able to push their own changes live. Both of these requirements conflict with the way Etsy typically operates. Here's how they approached PCI-DSS compliance.

1. Minimize the fallout of the required compliance. Understand there is no one-size-fits-all compliance solution, and architect systems to separate the concerns related to different compliance demands.

Etsy's mainstream engineering culture is optimized for speed of innovation. However, credit card processing is an area where user data security is paramount. Etsy recognizes that different parts of their system have different concerns and need to be treated differently.

Etsy's most important architectural decision was to decouple the CDE environment from the rest of the system, limiting the scope of the PCI-DSS regulations to one segregated area and preventing them from "leaking" through to all their production systems. The systems that form the CDE are separated (and managed differently) from the rest of Etsy's environments at the physical, network, source code, and logical infrastructure levels.

Furthermore, the CDE is built and operated by a cross-functional team that is solely responsible for the CDE. Again, this limits the scope of the PCI-DSS regulations to just this team.

2. Establish and limit the blast radius of frameworks and regulations.

Always start by asking, "What's the smallest possible set of changes we can make to our ideal architecture and culture while still achieving compliance with regulations we are subject to?" Then take an incremental, iterative approach to implementing and validating those changes.

For example, while PCI-DSS mandates segregation of duties, that doesn't prevent the cross-functional CDE team from working together in a single space. When members of the CDE team want to push a change, they create a ticket to be approved by the tech lead; otherwise, the code commit and deployment process is fully automated as with the main Etsy environment. There are no bottlenecks and delays, as the segregation of duties is kept local: a change is approved by a different person than the one doing it.

3. Use compensating controls.

It's essential to respect the outcomes the regulations are trying to achieve, while recognizing there are many ways to achieve those outcomes. For example, PCI-DSS allows organizations to implement "compensating controls"—a workaround designed to create the same outcome—where there is a legitimate technical or business constraint preventing implementation of a particular control.[8]

In the case of PCI-DSS, you should talk to your qualified security auditor (QSA) and acquiring bank to discuss possible alternatives to controls that have an unacceptable technical or business impact. For example, the deployment pipeline described in Chapter 8 and used by Etsy provides a powerful set of compensating controls that can provide an alternative to segregation of duties in their other systems.

The advantage of using lean principles and continuous delivery in product development is that it enables a fine-grained, adaptive approach to risk management. As we work in small batches and are able to trace each change to our systems from check-in to deployment, we can quantify the risk of each change and manage it appropriately.

The best way to achieve the objectives of good GRC is by embedding compliance and risk management into the daily activities of product teams, including systems and UX design and testing. As organizations move away from the command and control paradigm and GRC teams adopt a collaborative approach to risk management, we begin to value them as trusted advisors and experts in their knowledge domain. For many GRC teams, this requires a major shift in their roles, responsibilities, and behavior within an enterprise

8 *http://bit.ly/1v732EU*

organization. This is the move from a policing role to that of a contributing team member who is measured on the same outcomes as the product team, not solely a compliance perspective.

Conclusion

Good governance requires everyone to focus on discovering ways to improve value and provide accurate information on which to base our decisions. We start with leadership and direction from the Board and Executives, and rely on the ability of employees to embrace their responsibility to make good decisions at work. A culture of openness, trust, and transparency is required for good governance.

GRC structures and processes must be developed collaboratively by both GRC teams and the product teams that work day to day to deliver value to customers. By identifying the intent of the laws and regulations we must comply with, our GRC teams can collaborate with product teams to determine local approaches that fit best with improving value delivery. We start by exploring, with GRC teams, how we can minimize the negative effects of relying on restrictive controls through creative use of system architecture, process improvement, containment of scope, applying compensating controls, and leveraging new technologies. We can then exploit our learning to continuously improve our processes to provide both better governance and better outcomes for all stakeholders.

Questions for readers:

- How do your product teams view your current GRC processes? To what extent is your organization engaged in risk management theater?
- What actions do leaders take to develop a shared understanding of GRC language and frameworks throughout the organization?
- Do your GRC structures (policies, organization, and processes) prevent product teams from performing process improvement or require them to seek approval for any process change? If so, how might you support teams improving their processes while maintaining compliance?
- How might you enable GRC teams to collaborate with your product delivery teams as trusted team members throughout the value creation process?

O'REILLY®

Wait. There's More.

4 Easy Ways to Stay Ahead of the Game

The world of web operations and performance is rapidly changing. Find what you need to keep current at **oreilly.com/velocity**:

More Reports Like This One

Get industry intelligence in timely, focused reports written to keep you apprised of the current and trending state of web operations and performance, best practices, and new technologies.

Videos and Webcasts

Hear directly from some of the best minds in the field through free live or pre-recorded events. Watch what you like, when you like, where you like.

Weekly Newsletter

News happens fast. Get it delivered straight to your inbox so you don't miss a thing.

Velocity Conference

It's the must-attend event for web operations and performance professionals, happening four times a year in California, New York, Europe, and China. Spend three supercharged days with the best minds, companies, and people interested in the same things you are. Learn more at **velocityconf.com**.