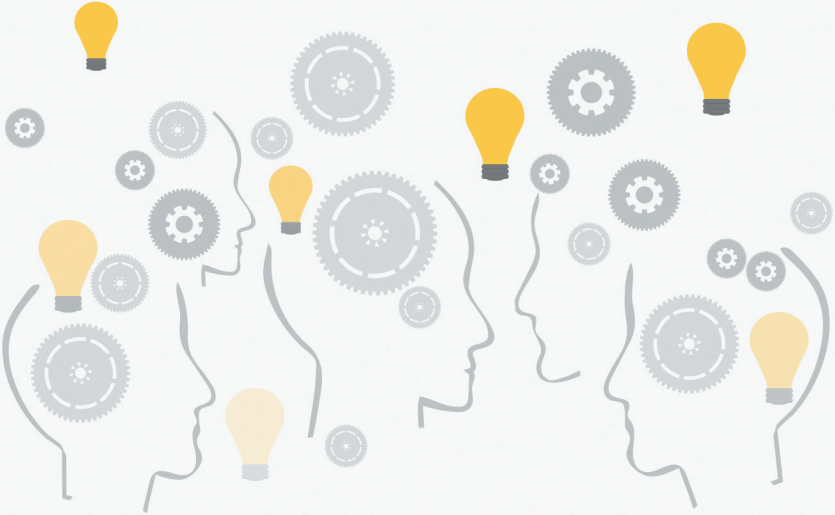
Deliver Better Results

How to Unlock Your Organization's Potential



GIL BROZA

FOREWORD BY JEFF GOTHELF

WHAT PEOPLE ARE SAYING ABOUT *DELIVER BETTER RESULTS*

This book is for executives who realize the system they lead is larger than just coders or a project management team, and who are looking to optimize the way their team works. It offers practical guidance without providing process recipes, being too philosophical, or treating employees and contributors as mere "resources."

~ Tal Reichert, CTO, Greenscreens.ai

This is a must-read for software development leaders and executives striving for excellence. Its insights make it a game-changer. Broza's brilliant emphasis on viewing software organizations as a cohesive system of value delivery hits the mark, and his 10 powerful strategies for optimizing value delivery are a treasure trove of actionable wisdom. As an R&D leader, I found the book's focus on incremental improvement and practical execution particularly refreshing.

~ Dinah Davis, former VP of R&D, Arctic Wolf Networks

Product people like frameworks but hate rules, and favor simplicity to complexity, but people + tech + orgs tangle up in complicated ways. This book unravels that knot with simple and clear advice that considers the whole messy human system, not just one department or specialty.

~ Andrew McGlinchey, VP Product, PropertyGuru, ex-Microsoft, ex-Google

Discover a people-first systematic approach in Gil Broza's groundbreaking book on technology product development and solution delivery. This book is your compass to achieving remarkable results in this ever-evolving space, blending strategic insight and practical steps in every chapter. It will transform the way you approach your role.

~ TK Balaji, CIO, Post Consumer Brands

So many leaders attempt to transform their organization by changing everything, but I've never seen that succeed. With Gil's book, leaders can now apply specific changes to create a more effective system of work. That's how to do a transformation of any kind.

~ Johanna Rothman, Consultant, author of *Manage Your Project Portfolio*

Clearly and concisely written, this book presents a solid foundation followed by practical improvement strategies for advancing the performance of an organization. Particularly inspiring to me is the treatment of humanity in our professional lives as an asset rather than trying to minimize it in the name of efficiency.

~ Christopher Marsh, former VP Engineering at Comcast

Gil's book provides a compelling guide for today's VUCA leaders looking for practical, systemic, people-first delivery strategies. Chock full of helpful advice, real-world stories, and context-based awareness, I can't think of a better guide for today's organizational leaders.

~ Bob Galen, Agile Coach, Agile Moose

Gil clearly understands the execution challenges that organizations face. On more than one occasion it felt like I was not simply reading a chapter in a book but receiving custom advice based on his personal observations. Since finishing the book, I've been more mindful of the overall system and pull from the contents often when meeting with leaders.

~ Alesha Foy, Director Enterprise Agility, BECU

This book is an invaluable resource for leaders who seek to foster a positive culture and build strong and effective product development teams.

~ Alon Sabi, Head of Engineering, Breadstack Technologies

Why does a great idea or design fail to result in a great product? It takes many different people, with diverse skills, collaborating effectively to create valuable products. This book gives you an actionable plan to break out of the silos that are holding your organization back from achieving its potential.

~ Jeremy Kriegel, UX Leader, Host of the Saving UX podcast

If you're looking to take your organization to the next level, this book is quite literally for you. Gil has masterfully created a simple, straightforward approach that will guide you every step of the way while giving you the freedom to adapt it to your unique context.

~ David Wallace, Principal Agile Coach, Xero

As a product leader, I find this book to be a refreshing departure from prescriptive approaches, offering instead a valuable set of insightful strategies and questions to guide both leaders and teams in unleashing their full potential. With a useful self-assessment tool as a guide, it's an effective resource for organizations seeking genuine improvement in how they deliver results.

~ Steve Rogalsky, VP Product Management

Gil introduces us to a tailored approach for greater business agility. Giving clear, step-by-step guidance and observable metrics, he helps the reader achieve optimum fitness for their organization. People using the Agile Fluency® Model have asked us: How do we achieve organizational fluency? Deliver Better Results has the answer to that question.

~ Diana Larsen, Leadership Agility Advisor, author,
Lead Without Blame and other books

Read this book today; start improving your delivery capabilities tomorrow. Gil Broza offers a simple and pragmatic model—grounded in real-life experiences—for assessing current state and introducing systemic change while keeping people firmly in the foreground. I'm looking forward to bringing this empirical and human-centered approach to the leaders I work with to help them address crucial delivery challenges right away, no matter where they're starting from.

~ Ellen Grove, Business Agility Coach, Agile Partnership

Deliver Better Results provides a motivating approach to hone the system you've always wanted at the scale you've always dreamed of!

~ David Johnson, Principal Software Engineer & DevOps Facilitator, Skillsoft

This book exemplifies having a "people first" approach. It explains in detail how to enable an entire human system inside a bigger complex context.

~ Malene Krohn, VP, Excellence in Product Development, Leadership, and Operations, SimCorp

Deliver Better Results is an essential read for forward-thinking leaders. It provides a transformative roadmap to effective product development and solution implementation by incorporating human-first insights with systems thinking. Regardless of your organization's approach—Agile, hybrid, or traditional—the strategies within these pages empower you to improve your value delivery.

~ Moe Ali, CEO, Product Faculty

This book strikes the right balance, giving you the steps to improve your organization while not prescribing a detailed formula that might not apply. Instead, it gives you strategies with examples from various companies and forces you to think about what would work for your specific situation.

~ Tim Grant, Sr. Technical Program Manager, Dejero Labs

A valuable leadership guide for exploring the bigger picture of systems and their impacts on delivery.

~ Tricia Broderick, Leadership Advisor, Ignite Insight + Innovation

The pragmatic guidance in this awesome book can be applied to any team or organization that wants to learn about their system of work and how to improve their outcomes incrementally.

~ Debbie Brey, Enterprise Agile COE Leader, The Boeing Company

ISBN print: 978-0-9880016-7-1
ISBN ebook: 978-0-9880016-8-8

Cover design: BookCoverExpress.com

Editing: StoriesRulePress.com

The publisher offers discounts on this book when ordered in quantity for bulk purchases or special sales, which may include electronic versions. For more information, please contact:

Gil Broza
(416) 302-8120
gbroza@3PVantage.com

Published by 3P Vantage Media

CONTENTS

INTRODUCTION

When I started my career in software development, the prevailing method of work was project-oriented and plan-driven. Later, other methods appeared, some becoming very popular. Each took a different approach to individuals and teams, to projects and products, and to processes and practices. Yet, each made the same claim: "This is how you'll deliver value successfully."

In reality, however, no company uses just one work method exclusively and exactly as prescribed. Every company's way of conceiving, developing, and delivering products and solutions is a mix of ideas from multiple methods, from management literature, and from its own people's brains. It's never perfect — and sometimes, far from perfect — so leaders try to improve it, whether gradually or via "transformation," with varying levels of success.

In 2021, I started to explore this question: "Even though all companies are different, what do all successful improvements to value delivery have in common?" My objective was to develop a simple model that leaders could use to improve their way of working whether it was product-oriented or project-oriented, plan-driven or agile or hybrid, based on a popular framework or home-grown.

The result of my exploration is a set of ten sequential and incremental strategies to apply across a value delivery system. They produce a real and sustainable increase in its fitness for purpose — meaning that the system better helps the company achieve its mission and objectives — without prescribing what form that must take. I based the model on my experience and observations from 30 years in the world of software development, the last 19 of which I've spent as coach and consultant to over 100 organizations of all sizes

and industries. I've iterated over it using feedback from dozens of senior managers and other consultants. Most importantly, I stand on the shoulders of giants too numerous to name here.

And now, I've captured the model in this: the minimum viable book for improving value delivery. Since there's so much to know and do on this front, this book could have easily been 1,000-pages long, and you probably wouldn't have picked it up. Instead, I've written it such that you only need to read a little to know *what to do next in your current situation and why*, without imposing specific choices on you.

The book provides just enough theory to inform your actions. It includes dozens of real-world examples from clients and leaders to inspire you. And, its guidance empowers you to act effectively *in your context*. You can download free supplementary resources, such as handy summaries and self-assessment questionnaires, from the book's companion website, DeliverBetterResultsBook.com. To dive much deeper into certain topics, you'll find some of the best references I know in the "Further Reading" appendix. You'll also find a list of questions for initial discussions with fellow improvement leaders.

I hope this book helps you navigate your improvement journey effectively and efficiently. I look forward to hearing about both your successes and challenges. You can reach me at gbroza@3PVantage.com.

Gil Broza, Toronto, 2023

CHAPTER 1
THE BIG PICTURE

When you have a few quiet minutes at work — or when there's trouble — do you sometimes wonder:

"What can we do to deliver better results?"

Perhaps you're worried that your products or solutions don't meet customer and business needs as well as they should. Or, you're frustrated that despite the team's hard work, they keep falling behind, and the business needles aren't moving enough. Or, things are okay, but you can tell that there's a lot of untapped potential.

And so, you might consider upgrading processes, adopting better tools, or restructuring teams. You might also wish to get better at your own role and responsibilities — to better lead people, evolve products, design software, coach teams, and so on. To achieve the impact you're looking for, the following three matters are critical:

1. Improvement efforts should focus on the entire system of value delivery.

There's an area of your company that creates technological products and solutions for the benefit of the company's customers. The benefit is direct if the customers are the technology's users, and it's indirect if fellow colleagues use the technology.

That area is an entire *system*. It comprises the team members, management, and ways of working involved in conceiving, making, and delivering the technology. As such, changes in one part may impact other parts or be offset by their behavior, and may not improve the whole. For example:

- Starting to run product experiments will be short-lived if management always requires detailed, months-out commitments and plans.

- Releasing product updates more frequently may increase business risks if the code is sometimes unsafe.
- Abolishing meetings in the name of productivity may reduce the quality of planning decisions.

It's the results of the system — not what its parts do — that matter to customers and the business. Therefore, delivering better results requires coordinated and aligned changes across the system.

For many companies, that's a challenge: they don't manage this area as a system but as several independent parts such as Product and Engineering or Business team and IT team. It also doesn't help that as an industry, we don't have a single, unambiguous name for it. Think about other company systems, such as Sales, Finance, and HR; from their names, you know immediately who and what they include. Not this one. Many professionals refer to it as Product Development, which some others interpret as only the coding and testing part. Others refer to it as Technology, Value Stream, Project/Product Pipeline, or Software Delivery Operations, all of which may be susceptible to limiting associations and interpretations. Perhaps you have a different name for it. In this book, I'll refer to it as "value delivery system," or "system" for short.

2. A model or theory should guide your system improvement choices.

It's not enough to set improvement objectives or to have frequent retrospectives. You need a model or theory of what improvement looks like and what changes will achieve it. In the worlds of product development and solution delivery, several rather different models exist, each with its advantages and disadvantages. Some, such as Scrum and its scaled versions, minimally implement a particular philosophy (Agile, in this case) and are silent on various system-level matters. Others, such as SAFe, prescribe a

specific target state for the system that may not be ideal for your particular company. And other models, such as the Theory of Constraints, suggest principles and actions for gradual improvement, but it's hard to know how the system would turn out afterwards.

3. A broad coalition of leaders should be supporting the changes.

Conceiving, making, and delivering value via technology is anything but straightforward and routine. In almost every company, it involves many interdependent people working hard to solve other people's problems within a complex and ever-shifting organizational and business context. Don't try making changes all on your own; to achieve real and sustainable improvement, collaborate with leaders from across the system. You need a coalition of leaders that, together, can:

- Overcome the organization's inertia, self-imposed constraints, and cultural barriers.
- Influence people, who have different world views, accountabilities, and concerns, to align to a set of choices so they may achieve shared success.
- Build environments where *people* — not so-called *resources* — show up and work together at their best.

FITNESS FOR PURPOSE

In thinking of value delivery as the product of a system, I've found it helpful to consider the system's **fitness for purpose**: how well it helps the company achieve its mission and objectives. By making your system more fit for its purpose, you'll deliver better results.

Having supported improvements in dozens of such real-world systems and studied many others, I've noticed that fitness for purpose generally corresponds to one of the following

five progressively better levels. It's the same five whether the system is product-oriented or project-oriented, plan-driven or agile, based on a familiar framework or home-grown:

> **Level 1:** the system has some successes, but is unable to contribute adequately to achieving company objectives.
>
> **Level 2**: the system contributes to achieving company objectives, but neither effectively nor efficiently enough. (Effective: doing the right thing, solving the right problem, achieving the intended goal. Efficient: doing so with minimal waste of time, effort, resources, money, goodwill, etc.)
>
> **Level 3**: the system's results are satisfactory, but fully dependent on a few people who make all the high-impact decisions.
>
> **Level 4**: the system is effective and efficient, but slower to achieve major outcomes than it needs to be.
>
> **Level 5:** the system produces all the results the company needs from it.

Systems that achieve great fitness get there one level at a time. They don't transform in one go into Level 5 systems: that's much more change than people can handle, even if the target state is clearly understood and communicated. In this book, we focus on going up just one level at a time, which isn't trivial either. It may take months and may be rocky, even in ideal conditions; some backsliding is not unusual.

While you might be keen (or pressed) to improve your system and deliver better results, you might also be experiencing a conflict. Amid the demand and flux of business, there never seems to be a good time to make changes. And when you do contemplate changes, so much advice on improving value delivery is available these days, it quickly gets overwhelming. At the same time, maintaining the status quo involves costs and risks.

This begs the question: What's an efficient way to level up reliably?

I've found that given a system's current fitness level, **two or three specific cross-system strategies will move it up one level effectively, efficiently, and sustainably**. In total, there are ten of these incremental and sequential strategies for moving from low to high fitness — from Level 1 to 5. They apply to all systems, though naturally they need to be customized to each system's parameters and unique context.

In this chapter, you'll determine the scope of your system, assess its current fitness, and learn briefly about the strategies that will take it to the next level. In the rest of the book, you'll learn about the foundation of leadership that makes these strategies effective and sustainable, and dive deep into each one. All of this, taken together, forms a model I've developed called **SQUARE** (because it's designed to be **S**imple, **QUA**litative, and **RE**lative).

DETERMINE THE SCOPE OF YOUR VALUE DELIVERY SYSTEM

Before you start applying this book's guidance to your system, make yourself a clear mental picture of its scope — who and what it includes. Use the following perspectives to think about it.

- **People and work:** The system includes all the individual contributors *and managers* involved in the product/solution from idea to delivery, their interactions, their work, and their methods and tools for getting work done. Its people — employees, contractors, vendor staff — may work on different teams and report to different managers (not only in technology), but they're all interdependent and necessary for conceiving, making, and delivering a complete product. That product matters to external users and customers,

and/or to internal people whose work benefits external customers.

- **Parts:** The system is likely made up of parts. For example, if you work on a product line in a tech company, the parts are likely to be product management, design, engineering, and delivery, among others. If you develop software for internal purposes, the system is the equivalent in terms of IT and their business partners. If your company provides software services to other companies, the parts are development, analysis, project management, account management, client representatives, and others. Your company *might be used to managing all these parts separately* — perhaps as functions — but none of them can achieve customer and user outcomes and move business needles on their own; the system does that through all its parts working together.
- **Boundaries:** The system is a distinct subset of the company. A lot of people may care about the product/solution — or even suffer if it's poor — but unless they have some influence or control over its evolution, they are not part of the system that makes it. This is the situation, for example, in some companies with respect to marketing, HR, and customer support; in others, representatives of these functions have some influence over product choices, and are therefore part of the system.
- **Multiple systems:** Your company may have multiple value delivery systems (which might intersect somewhat). For example, in some companies, one system's aim is to explore new opportunities, and another's aim is to exploit existing opportunities. Apply this book's guidance to each one separately.

For an additional perspective, imagine that the product is a movie. What and who would the credit roll include?

Example: A company developing a marketplace in the mobile app space had two systems. One focused on the company's own app for mobile users, their engagement and retention, monetization, and app discovery. The other focused on providers of marketplace apps, ad and bidding tools for them, fraud, and more. Each system included several teams, every team consisting of front-end and back-end developers, testers, product managers, data scientists, and data analysts as their mission required. Each system also included team leads, managers, and directors. Three senior leaders were in both systems: the heads of engineering, product, and data. The CEO and the COO operated outside both systems: they did not control the workings of either one, though their decisions naturally affected the choices that system leaders made.

ASSESS YOUR SYSTEM'S CURRENT FITNESS FOR PURPOSE

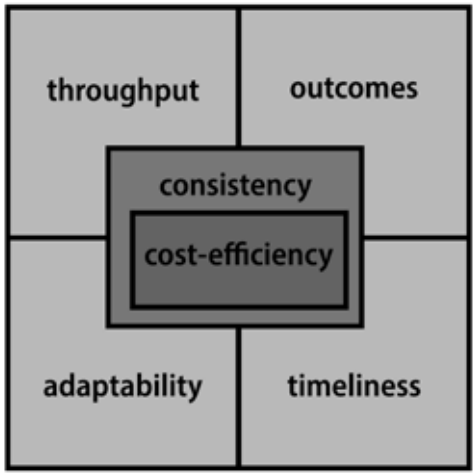
The fitness for purpose of a value delivery system is how well it helps the company achieve its mission and objectives. This is a *relative* concept, since the mission and objectives vary from company to company. SQUARE defines six aspects of system fitness and a simple, qualitative way for rating them and subsequently assessing the system's fitness level.

For an analogy, think about physical fitness. If there were some universal, absolute scale for measuring a person's fitness, yours would likely be a fraction of that of an athlete who trains daily for the Olympics. However, the *purpose* or goal for which you need fitness may be very different from the Olympian's: enjoy runs in the park, volunteer as a firefighter, or climb mountains. And then, consider the *aspects* of physical fitness, such as strength, speed, endurance,

and agility. Each of these aspects matters differently given the purpose.

To rate how your system does on each aspect, you'll work through the same three questions: what's the practical and relevant optimum for that aspect? What's the current state? How is the current state relative to its optimum? I recommend you write down your answers, at least to the third question. I've provided real-world examples for rating each aspect, as well as two examples of complete assessments. That's why this section ended up rather long!

As you'll see, this assessment is subjective. Try to be neutral, but critical; fair, but not harsh. Base your answers on actual results and behaviors, not on how people talk about them. If you struggle with a question, consider this angle: how would an independent outsider, who knows your system well, answer it?



The six fitness aspects in SQUARE

Throughput

The first fitness aspect is **throughput**: the amount of usable product/solution delivered by the system in short spans of time. (Note: *usable,* not *useful.* We'll get to useful in a moment.)

Question 1: What's your system's *optimum* throughput?
In other words, what is the throughput like when the system is most fit for its purpose? What throughput (as defined here) would best serve the customers and the company?

The **optimum** must be **practical**. The system operates in a specific business and technology landscape with constraints and obligations. Even with ample time and funds, there's a limit to what would be practical.

The optimum must also be **relevant**. Even if you draw inspiration from industry stars, what throughput would *matter* most to your users? What throughput would have the best net effect on the system itself, for instance in terms of risk, learning, or adaptation?

The optimum doesn't have to be extreme. Keep in mind too that in technology development, more output (features, redesigns, enhancements) doesn't always equate to higher value or better quality.

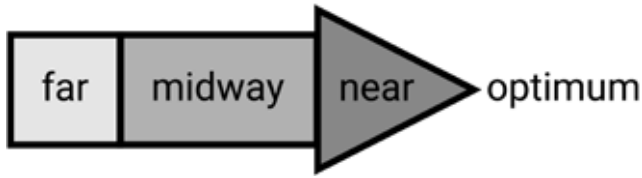
If you're struggling to determine the optimum, try this. Given what you know of the industry, consider what the ideal throughput would be for a company like yours. Then dial it down based on your system's constraints: culture, budget, staffing, regulations, technology, etc.

Question 2: What's the system's *current* throughput?
Consider the last few months. The system's throughput would have varied over that time and across teams, so think about its average. A rough idea will do; no need for a precise answer.

Question 3: How is the current throughput *relative* to its optimum?
Rate the current throughput as follows:

- If it's far enough from the optimum that it is (or should be) a constant point of concern for management and stakeholders, rate it as "**far**."
- If it's near the optimum enough that it's not an issue (it's good enough), rate it as "**near**."

- If it's neither near nor far, rate it as "**midway**."



Examples:

- At a company that made a 3D visualization product, monthly releases would have been practical and most valuable. However, releases were done only every six months (at least), which wasn't good enough for the company's needs. Throughput was "far."
- A team working for a university was responsible for an established product for researchers. The team released small enhancements every week, which was just right both for them and for their customers, so throughput was "near."
- At a company that built software for product management, the development of new features took longer than desired (due to team size and aging infrastructure), but not so long as to be a constant problem, so throughput was "midway."

Note: If the product/solution hasn't had its first release yet, base the optimum (for this aspect and subsequent ones) on what's ideal for bringing it to customers for the first time. For instance, say that four months ago you started developing a large new product, and it was ideal to ship an early-adopter first version within a year. Also, say that the optimum throughput is that every two weeks, more basic functionality is completed, because that reduces risk. If that's in fact happening, throughput would be "near." However, if the teams are still debating the architecture and not validating it, throughput could be "far."

Outcomes

The second fitness aspect is **outcomes**: the system's achievement of valuable customer and business outcomes — solving problems, addressing needs, achieving goals, seizing opportunities — both large and small. This aspect also covers the matter of product/solution quality, since quality — as the customers/users see it — is what makes the achieved outcomes particularly valuable to them.

Work through the same three questions as before, this time applying them to the system's achievement of outcomes:

1. What's the practical and relevant optimum?
2. How does the system currently perform?
3. How is the current performance relative to its optimum — far, near, midway?

Examples:

- At a financial institution that was modernizing its self-serve investment platform, the teams sequenced and evolved features based on data-driven analyses of user journeys; outcomes were "near."
- At a retailer undergoing a digital transformation, teams regularly started, changed, and abandoned features, because management couldn't decide which ones would matter the most. Outcomes were "far."
- At an educational game maker, teams regularly delivered useful features (so, outcomes weren't "far"). However, based on business metrics, those were not always the best features to work on. Therefore, outcomes weren't "near" either; they were "midway."

As with throughput, you might notice differences across teams. For example, one product manager might focus her energy on outcomes and validating hypotheses, while other

product managers focus primarily on populating backlogs with features. Think about the *entire* system's achievement of valuable outcomes.

Timeliness

The third fitness aspect is **timeliness**: the system's delivery of outcome-producing results when they're still valuable enough. This isn't about hitting company-determined deadlines; rather, it's in the eyes of the customers/users (who might be internal): by the time they receive updates or solutions that address their outcomes, how valuable are they?

Now work through the three questions:

1. What's the system's optimum for timeliness? Remember that it should be practical and relevant.
2. What's the current performance with respect to timeliness?
3. How is the current performance on timeliness relative to its optimum — far, near, midway?

Examples:

- In addition to regularly enhancing its digital products, a large media corporation had to customize them for major world events. The teams delivered enhancements on good schedules and their customizations were always ready for the events; timeliness was "near."
- A company had a product for connecting people, but its technology and UI were quickly getting obsolete. Three teams were busy rewriting them to current standards and expectations, but their progress was much slower than needed (due to various technical and leadership reasons) and the initial launch kept being pushed out by months. Timeliness was "far."

- The financial institution mentioned earlier deployed a bare-bones version of its new investment platform and then added features gradually. Customers didn't have to wait ages to experience its benefits, but the development of most features took a long time. Therefore, timeliness was "midway."

Adaptability

The fourth fitness aspect is **adaptability**: the ease and speed at which *both the system and its product* adapt to important changes. Changes may be due to internal choice or to external conditions, and adaptations may not necessarily be net positive.

Work through the three questions:

1. What's the optimum adaptability for both the system and its product? Its relevance is particularly important: the world moves fast these days, but not equally for everyone.
2. What's the current state of adaptability?
3. How is the current adaptability relative to its optimum — far, near, midway?

Examples:

- The university team noted earlier kept a small roadmap of features, which they reviewed frequently and adjusted as needed. The product's architecture was already proven, and changing features was relatively easy. Therefore, adaptability was "near."
- The company that made the 3D visualization product used to make big annual plans, which were hard to change. As well, the teams couldn't change the product quickly and safely for various technical reasons. Adaptability was therefore "far." After some coaching, management moved to six-month roadmaps of sequenced outcomes

(with suggested but not binding dates), and the system's adaptability changed to "midway."

These four aspects are the foundational ones, because the next two "go meta": each one relates to the previous ones.

Consistency

The fifth fitness aspect is **consistency**: the system's continued achievement of its throughput, outcomes, timeliness, and adaptability. Naturally, it varies from week to week and from month to month; the lower the range of variation, the higher the consistency.

Repeat the three questions, this time for consistency:

1. What's the optimum?
2. What's the current state? Your answer here should reflect the system's normal and sustainable operation, so if it's recently been in a short crunch mode, base your answer on how things were before it.
3. How is the current state of consistency relative to its optimum?

Examples:

- At the company that was rewriting its product for connecting people, the new version's features, design, and architecture were constantly in flux. Some sprints the teams delivered useful features, and the next sprint they were told to change them. Sometimes the product seemed to be getting close to market-readiness, and then it wasn't. Consistency was "far."
- The media corporation mentioned earlier had an appropriate balance of planning and adaptation, and teams regularly built and delivered useful features. Consistency was "near."

- At an investment company, business people populated large backlogs that teams then processed in sprints. Throughput and timeliness were usually "midway," whereas outcomes and adaptability were sometimes "far," sometimes "midway." On the whole, consistency was "midway."

Cost-efficiency

The sixth fitness aspect is **cost-efficiency**: the efficiency of achieving the system's current throughput, outcomes, timeliness, adaptability, and consistency for the money spent. It aggregates everything paid to employees, contractors, and providers to make the system produce its effects on these five aspects over time. That includes costs for new development, maintenance, process administration, tools, licenses and services, onboarding and training, and more.

One last time for the three questions:

1. What's the optimum? Be careful here: though every company can always cut some costs — and it might be forced to by external conditions — that doesn't mean it's inefficient *for the effects it's producing*. Furthermore, some kinds of cost-cutting may actually compromise the system's performance.
2. What's the current state?
3. How is the current state relative to its optimum?

Examples:

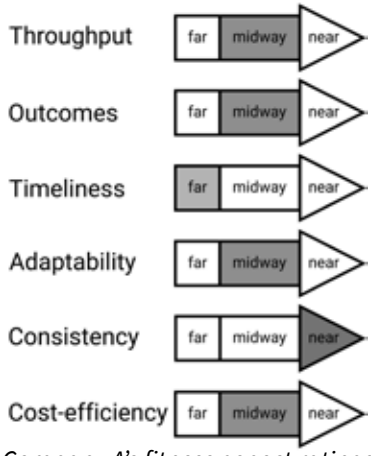
- At the educational game maker, teams had great throughput and adaptability and they did well on the other three aspects. However, everything was planned afresh every quarter, which often resulted in throwing away previous-quarter work that wasn't yet releasable. Cost-efficiency was "far."

- At the company that built software for product management, the costs were right for the system's performance. Cost-efficiency was "near."
- At the investment company, cost-efficiency was "midway": the system was right-sized for its needs and accomplishments, but it relied on many contractors and the process was heavy.

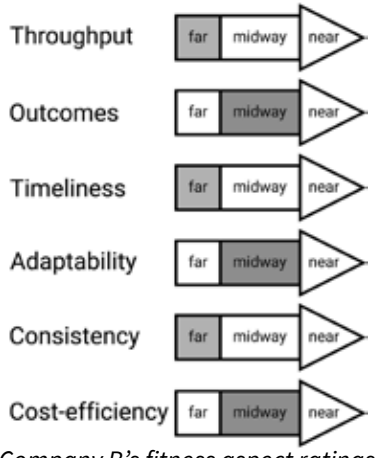
Here are two examples of complete assessments:

1. Company A, a fast-growing tech startup, had five Scrum teams working in two-week cycles, serving both public users and an internal services organization. The teams regularly made good progress, but the company needed more for its growth targets, so Throughput was "midway." Outcomes were also "midway": although product backlogs were generally prioritized by value, management was frequently nervous about the choices made. Delivery of complete and customer-meaningful features generally took long enough to rate Timeliness as "far." Changing direction in the product and adjusting the way of working weren't painful, but they weren't always easy either, making Adaptability "midway." The support from Product and Engineering directors enabled the teams to keep a good rhythm, making Consistency "near." Cost-efficiency was "midway," mostly due to the type and extent of technical debt.

2. Company B was a veteran provider of services to diverse public sector organizations. About 70 people in a digital solutions department worked on multiple customer-specific and platform-enhancement projects. The high number of concurrent projects and extensive matrixing of people resulted in a lot of late deliveries and escalations; Timeliness was "far." Throughput was also "far": although project teams nominally worked in sprints, they avoided releasing code frequently because it was hard to ensure quality. The products fulfilled their users' goals minimally and usability was basic, making Outcomes "midway." Changing or extending current products (such as due to new contracts or laws) was hard but not a common occurrence, so Adaptability was "midway." Consistency was "far": It was impossible to know whether the next month was going to go well or not. Even though the department was kept small and the teams were fully loaded all the time, the overhead of constantly coordinating and redirecting staff meant cost-efficiency was not "near" but "midway."



Company A's fitness aspect ratings



Company B's fitness aspect ratings

Having produced your ratings for all six aspects, I suggest you double-check them:

- For each aspect, did you think of it as defined here and rate it independently of the other ones? For example, if you thought "We deploy product releases twice a year, but we should deploy much more frequently than that," throughput can be "far" while consistency is "near" (and the system might be quite cost-efficient for its release cadence).
- If your analysis revealed that the system optimizes for some aspects at the expense of others, that's useful information — it doesn't mean you've rated them incorrectly. In a common example, a system that changes its direction and focus excessively may rate highly on adaptability but poorly on cost-efficiency due to all the abandoned work.

Consider inviting fellow leaders to perform this assessment independently, and then meet to compare your ratings. If theirs are similar to yours, that's a good indicator of alignment. If the ratings differ materially, ask everyone to articulate their views of the aspects' optimums and then of the current state; remember that the optimums need to be both practical and relevant. Either scenario will make for insightful discussions.

Now, calculate the level of your system's fitness for purpose:

1. Convert each rating to a numeric value as follows: far = 1, midway = 2, near = 3.
2. Sum up the numbers to produce the **raw fitness score**, which should be between 6 and 18. (Note: This is a simple sum. Don't prorate the ratings or assign a weight to each aspect.)
3. Find your system's fitness level in the following table:

Level 1	Level 2	Level 3	Level 4	Level 5
6-8	9-11	12-13	14-16	17-18

Keep a record of your six ratings and the raw fitness score, so you can discuss them with colleagues and track changes over time. However, the level you've just calculated will be enough for knowing which improvement strategies to employ.

If the level is lower than you'd like it to be, take heart. Acknowledge the successes and positives in your journey so far, and read on to discover what will help you deliver better results.

SUPPLEMENTARY RESOURCE: Download "Fitness for Purpose Self-Assessment Questionnaire" (the above assessment in questionnaire format) from the book's companion website, DeliverBetterResultsBook.com.

HOW TO IMPROVE FITNESS

If you've rated an aspect or two as "far" or "midway," you might be thinking at this point: "Why don't we just put all our efforts toward fixing that?" For example, if outcomes are not great, our product managers and designers can run more experiments; if timeliness is poor, let's double down on correct estimation, detailed release and sprint planning, and frequent status checks.

Doing this may not actually produce the result you hope for. The reason, which Chapter 2 explains in much more detail, is that you're dealing with a complex system. Every change requires certain system-level readiness, changes in one part affect other parts, and improving one aspect may compromise another. To use the aforementioned examples:

- Running experiments regularly takes more than product thinking and a technical infrastructure. Leadership's approach to making product/solution commitments, planning, and even managing employee performance can make or break experimentation.
- Trying to make team planning and estimation more correct requires the teams to be effectively staffed for their mandate and to not be overwhelmed by many competing priorities. It may also reduce adaptability and cost-efficiency.

SQUARE's ten strategies don't target individual aspects or prescribe a process. Instead, *each strategy makes specific, principled changes to the* ***way of working*** *of the whole system*: how people engage with each other, decide what to work on, make commitments, plan activities, make choices while working, and deliver value. That improves all fitness aspects, though to different extents.

These strategies are **sequential** and **incremental**. I've found this to be practical, sustainable, and respectful of human systems' appetite and tolerance for change. If your organization is contemplating a "transformation," these strategies are an effective way to execute it gradually. Using even one of the strategies indicated for your system's current level will help you deliver better results; you don't have to be pursuing perfection.

Sequential: Implement the strategies in the sequence given and don't jump ahead. For instance, if your system is at Level 2, implement only that level's strategies and don't worry about those for Levels 3 or 4. Applying the later-level ones now would either not improve fitness much or be too difficult to pull off. Moreover, it would mean *a lot* of simultaneous change, and a possible distraction from the strategies that do apply now. Just keep the progression in mind so you don't undermine it inadvertently.

Incremental: To move from one level to the next, you need to execute only two or three strategies to a point where their adjustments are sustained and their effects are real. Expect that to take considerable time and effort! With supportive leadership, appetite and tolerance for change, and expert guidance, you'll minimize the time and be able to execute them in close succession.

Whatever your system's level, and whether you're currently trying to improve it or waiting for the right opportunity, **make sure to keep the lower-level strategies in play**. That should be easy if you've been following the book's advice since Level 1. However, if your system is at a higher level and you're encountering SQUARE for the first time, read up on the lower-level strategies and be sure they're being applied effectively. That is critical both for maintaining the current fitness and, if you're actively trying to level up, for getting maximum benefit from the new strategies.

To be effective, these strategies require a **foundation**: intentional leadership that actively builds and protects an explicit culture and way of working throughout the system. Chapters 2 and 3 explain and provide guidance for the specifics of that foundation.

Lastly, be aware that your system always faces multiple risks to its continuing performance and fitness. At each level, the highest risk is different, and though the risks from lower levels have likely been mitigated by the execution of those levels' strategies, they are still present.

THE 10 STRATEGIES FOR LEVELING UP

This section provides a brief summary of each level and its highest risk and then a one-paragraph description for each of its strategies for leveling up. It also provides advice for remaining at Level 5. Chapter 4 describes a pathway to follow when applying a strategy, and the subsequent chapters provide detailed guidance for each one.

It's important that you read the following subsections all the way from Level 1 to your system's current level. Also, if your system's raw fitness score is at the top end of the corresponding level (for instance, a raw score of 11 is at the top end of Level 2), be sure to read the next level — it's coming up soon.

Remember that the scope of application of each strategy is the entire system as you identified it earlier — not a single function or team and not the whole company either.

Progressing from Level 1 to 2

At this level, the system is unable to contribute adequately to achieving company objectives. The team works, sometimes very hard, and they have some successes. But on the whole, their deliverables fall short of what the business needs.

Often, the main culprit appears to be the team's process. That might be correct, but there's usually a deeper problem at the portfolio level: too many big-ticket items (projects, features, initiatives, etc.) are in flight or about to be. Various stakeholders, trying to accomplish their objectives, push work into the system and all of it fights for attention. The team tries to satisfy all the demand. Management spends considerable effort on prioritization, coordination, and escalations, but the system can't keep up.

This is usually a tense situation, because people — competent and dedicated as they are — feel that they can't succeed. *The highest risk to the system is that the key people who keep it operational may decide to leave.*

Strategy 1a: Manage the project portfolio with greater strategic control over committed and in-progress items.

Limit, choose strategically, and properly frame all portfolio items. (A portfolio item may be a feature, feature set, "epic," technical initiative, experiment, maintenance project — something big enough to make a demonstrable difference to the customers/users and the company.) Placing self-imposed limits on the number of portfolio items will clear

some of the logjam and enable the system to get more work to a deliverable state more often. Strategically prioritizing the exact items to work on will further improve the system's results. This strategy requires real accountability from the decision-makers, since their choices — and how frequently their choices change — have a large effect on effort, focus, and complexity, and therefore on company results. As well, this strategy often requires the influence and involvement of very senior people, because it's hard to say "no" or "not yet" to important work, especially when it's tied to managers' objectives.

Strategy 1b: Design the way of working based on what matters most for achieving the mission and objectives.
The operating model needs to be capable of producing the system's intended results. Rather than adopt a favorite/familiar/popular model and try to make it work in your context, design yours intentionally and explicitly. Start by getting leader alignment and consensus about what the system needs to be like if it's highly fit for its purpose. From there, work together to answer these questions: What is vital for the system to deliver the right product/solution to the right customer at the right time, and why do we choose the answers we do? Use the answers to determine the way of working. For example, if frequent delivery and high adaptability are vital for product success, you'll probably organize around semi-autonomous teams working in flow or sprints.

Progressing from Level 2 to 3

A Level 2 system addresses the needs of customers, stakeholders, and management but not effectively and efficiently enough. As a result, some of those needs produce interference, unplanned work, scope creep, frequent priority changes, escalations, and aborted work. These disruptions are *the highest risk to the system's continued fitness: the structure and process, which are already not optimal, are constantly under*

threat. Often, this triggers process patches and restructurings that satisfy a few imperatives at the expense of overall system fitness, risking a drop to Level 1.

Strategy 2a: Establish clear and appropriate decision-making.

Whether you've organically evolved your way of working or adopted off-the-shelf process frameworks, chances are there's a gap: who makes some decisions, when, and how is unclear, inconsistent, or not fully defined. Closing this gap is a necessary, though not sufficient, condition of both effectiveness and efficiency. To do that, review every type of high-impact, product-affecting decision, and gain clarity and acceptance on which individuals or groups make it and how. Use the system's intended values and principles, as determined in Level 1 and perhaps refined since then, to guide the choice for each decision. For example, if the system operates according to Agile values, many decisions would likely be made collaboratively by cross-functional groups; if it operates in a top-down functional model, many decisions would likely be made only by functional and project managers.

> ***Terminology note:*** **I define "Agile" as any way of working that is *congruent with the spirit* of the Agile Manifesto. It may factor in modern insights and apply across more kinds of work than software development.**

Strategy 2b: Stabilize the system.

Create an acceptable and sustainable balance between the variable demand that the system receives and the supply it produces. Be wary of common practices that may help some, but when executed regularly compromise various fitness aspects: making complete detailed plans, preparing precise estimates, maximizing people's utilization, and working long hours. Instead, use principles and tactics that improve the

flow of work. Capture all the work in a rich visual form; that will help you finish what's started, reduce unplanned work, and deal promptly with bottlenecks and high-variability delays. Prevent "clogging the pipe" by constraining work intake, breaking work down, and keeping spare capacity. On the human front, cultivate learner safety, increase trust, and enable people to contribute outside narrow specialties. Collect process data, but use it only for system improvement; be careful not to let data collection, analysis, and subsequent actions compromise psychological safety and trust.

Progressing from Level 3 to 4

The typical Level 3 system contributes satisfactorily to the achievement of company objectives. Usually, only one or two of the six fitness aspects are of ongoing concern, while one or two others are close enough to their optimum. If the operational model is based on short cycles, the teams churn out working product on a rather even keel; if it's more project- or date-driven, teams generally hit their dates.

Look closely, and you'll notice why the system is only at Level 3: its results are fully dependent on a few people who make all the high-impact decisions. Typically, those folks are product leads, architects or senior individual contributors, and middle managers — and they don't act as a team. Some might be considered "heroes." Two unvoiced assumptions are shared throughout the system and explain a lot about it: the planned work equals the right work, and the potential improvement gains from collaboration aren't worth the time.

The overweighting of those few people in decisions about the work, and the teams' unwavering cadence of delivery, lead to disengagement. Team members, instead of tapping into their creativity, feel like cogs in a machine — fully loaded "resources" that check tasks off, disconnected from the mission and/or their customers. The machine, however, is optimized a certain way, and has low capacity to adapt to big changes. *The highest risk to the system's fitness is*

chaos and breakdown due to big changes. Sometimes, the key people mentioned above are the only barrier to the chaos.

These phenomena may create a vicious cycle: the less team members engage (the more they "just do the work"), the more the decision-makers operate in isolation. Such systems are likely to retreat to Level 2 if enough key people leave.

Strategy 3a: Increase contributor safety, real teamwork, and collaboration.

This multi-component strategy sits squarely on the human side of the system. Work with leaders on creating contributor safety, one side of which is the safety to engage where others are involved: to offer different perspectives, to take initiative, to deliver bad news. For the other side of safety, upgrade the process so folks may do their work without fear of failure or trouble. Check if teams are really teams or only workgroups; if it's the latter, find and address the root causes that real teamwork hasn't materialized. Look for situations that would benefit from collaboration, and create the conditions that make collaboration possible and welcome both within teams and across the system. By unleashing people's potential and synergy, this strategy mitigates the key-person risk and improves results and resilience. It's also an enabler of the next two.

Strategy 3b: Defer commitments and increase release frequency while controlling costs.

In choosing what work to do, the system looks ahead to different horizons, for example 12-24-month roadmaps, 3-6-month releases, and 2-week touchpoints. It makes commitments whose nature and extent vary by horizon. The first component of this strategy is to commit to less and plan in less detail for the longer horizons. The second component, which will now be easier, is to increase the frequency of releasing product/solution updates. As you do all this, prevent two important costs from rising too much: the cost of change (that is, the affordability of likely future

adaptations) and process costs (planning, coordinating, releasing, etc.). Implement this strategy gradually because commitments and release frequency affect other systems in the company and are affected by them, and therefore require greater alignment and partnership with all those systems' leaders. This strategy will result in risk reduction, earlier value to customers, easier pivoting, less wasted work, and greater resilience to big changes.

Strategy 3c: Engage teams meaningfully, collaboratively, and efficiently in planning.

People in your system engage in a lot of planning: all the activities that lead to commitments, what they encompass, and how the system will act on them. Examples abound: product discovery, solution design, work breakdown, release and sprint planning, scheduling, and migration planning. Improve each activity's output by engaging the relevant teams and experts (in other words, not only the decision-makers or managers). Experiment with ways for them to contribute to the planning as meaningfully, collaboratively, and efficiently as possible. Much of this will happen in meetings; make them worth having and facilitate them well. As a result of this strategy, the system may produce better outcomes, achieve higher throughput (by reducing unnecessary work), deliver more-timely results (by reducing delays), and respond better to events.

Progressing from Level 4 to 5

The typical Level 4 system is both appropriately adaptive and a reliable producer of deliverables. Both staff and management see it as a good place to work. While on a tactical level there seems to be healthy progress, a higher-level view reveals why the system is not at Level 5: it struggles to pull off "big work" of strategic value to customers and the company, such as integrating an acquisition, re-platforming, or supporting new customer journeys. It succeeds eventually, but it's slower

than the company needs it to be. Therefore, *the highest risk to the system is a breakdown of process and good habits due to loss of stakeholders' trust and/or management's patience.*

Strategy 4a: Expand team ownership of major outcomes.
Increase teams' ownership as much as necessary and possible for the system's optimal fitness. Enable teams to experiment with their own ideas for better ways of working that would benefit the whole system; this might include re-teaming or starting up guilds and temporary task forces. Give them a greater say over identifying and sequencing major outcomes and how to achieve them. That will counteract the tendency to concentrate "big-work thinking" in the hands of a select few, who then present final-looking but too-big requirements to the technical teams. The extra brainpower and thought diversity will open up options for better solutions, while the increased participation will encourage the teams to put more heart and energy into work and reduce attrition.

Strategy 4b: Improve the inputs to decisions and the decision-making processes.
The previous strategies would have improved the quality of decisions made about the system and the product. However, some decisions still end up producing negative *actual* outcomes, sabotaging the *intended* positive ones and creating other problems (which may be hard to trace back to the decision). Improve the inputs to decisions by better understanding the customers (both external and internal), collecting feedback, and running tests, studies, and experiments. Improve decision-making processes by considering both intended and actual outcomes, using system thinking, and accounting for the cost of change. Enable your people to do all this by cultivating humility and challenger safety, reframing decisions as bets, and making decisions collaboratively.

Strategy 4c: Reduce the technical cost of change.
Your system and product/solution undergo change all the time, and it costs you. By implementing the strategies for Levels 1-3, you've been controlling the cost of change on several fronts except one: how much the system spends on technical modifications to the product. Some of those are infrequent, such as modernization, enhancing once-optimal solutions that no longer are, and correcting problematic past choices. Others are made practically every day as part of regular development (even if the team is building brand-new features). To the extent the time spent on all this change exceeds its ideal minimum, it diminishes the system's fitness. This strategy has two components. One: reduce the cost of the likeliest changes to *existing* parts of the product by making it cheaper, easier, and safer to work with those parts. Two, develop *new* parts in a way that makes the likeliest changes affordable. Day to day, that requires following technical agility principles such as rapid feedback, small and safe steps, and clean code. To make this strategy possible, three parties — development, product/business, and management — need to make an intentional, explicit, and mutual commitment to investing in the system this way.

Remaining at Level 5

A Level 5 system is uncommon: all fitness aspects are close enough to their optimum that there's rarely any issue. It takes years of deliberate leadership to cultivate such a way of working and the strong culture that enables it. Keep yours at Level 5 by ensuring that the ten strategies mentioned earlier continue to be in play throughout the system.

Despite the system's great performance, its *culture* might experience friction with that of the company. At the time of writing, this friction tends to occur in systems based on philosophies such as servant leadership, agility, and Lean that operate within companies that default to hierarchical control and predictability.

The friction is usually felt at the edges of the system, where it interfaces with the rest of the company. Senior people at the edges (such as VPs and product leads) buffer the system from the friction, managing it well through trusting relationships with their colleagues outside the system. *The highest risk to this status quo is a change at the top.* A new senior leader — at the helm of the system or just above it — may try to reshape it based on their own world-view and experiences in other contexts. I'm familiar with several cases where the cultural disruption was powerful enough to undermine the implementation of the strategies and send the system back to a lower level of fitness.

Two partly overlapping actions provide some "insurance" against disruption to the system's culture. One, continue building trusting relationships with stakeholders and executives. Hopefully, by now they see your value delivery system as a critical partner (not as an internal vendor) but you shouldn't assume that results speak for themselves. Two, help leaders of other company systems be more successful. Partnering, sharing lessons learned, and teaching them what you've learned along your journey will strengthen your relationships with them; to the extent they choose to shift their culture and ways of working to resemble yours, there'll be less friction.

Summary of the Five Levels

Level	Fitness	Highest risk	Strategies for leveling up
1	Has some successes, but is unable to contribute adequately to achieving company objectives	Loss of key people	Manage the project portfolio; design the way of working
2	Contributes to achieving company objectives, but neither effectively nor efficiently enough	Excessive disruptions → setbacks	Sort out decision-making; stabilize the system
3	Results are satisfactory, but fully dependent on a few people who make all the high-impact decisions	Big changes → chaos and breakdown	Increase safety, teamwork, and collaboration; defer commitments and increase release frequency; engage teams in planning
4	Effective and efficient, but slower to achieve major outcomes than it needs to be	Loss of patience and trust → breakdown of good habits	Expand team ownership; improve decision-making; reduce the technical cost of change
5	Produces all the results the company needs from it	Change at the top → being reshaped	Build trusting relationships; help others improve their systems

SUPPLEMENTARY RESOURCE: Download "Summary of the Five Levels" from the book's companion website, DeliverBetterResultsBook.com.

FITNESS DOESN'T STAY CONSTANT

Even when you're not actively looking to level up your system's fitness for purpose, expect it to fluctuate with time. The aspects' optimums may change, for example due to shifting market needs and preferences. The aspects' current state may also change, for example if enough key people leave or take on different responsibilities, or if technologies you rely on are retired. These changes might result in a shift to the ratings; for instance, an aspect might go from "midway" (okay but not great) to "far" (a real problem). For this reason, it's helpful to recalculate the raw fitness score every now and again and watch for changes.

A familiar special case of such change is rapid growth. Scaling up is often accompanied by increased expectations for the foundational aspects' optimums. However, quickly adding many new hires may send the performance on those aspects in the wrong direction: throttling throughput, compromising outcomes, reducing timeliness, and hampering adaptability.

If your system is growing fast or if its fitness has dropped, redouble your execution of the strategies that are already in place, starting with Level 1 and working sequentially to the current level. For example, an EdTech (education technology) company at Level 4 increased headcount by 20% to support its expansion to new markets. It had to update its portfolio management to reflect the new capacity (Strategy 1a) and to ensure that the team structure was appropriate (1b); then it had to plug newly formed gaps in decision-making (2a) and stabilize the bigger system (2b); then it had to "relearn" how to defer commitments (3b). However, doing all this was much quicker than getting to Level 4 the first time around.

WHAT TO READ NEXT

You've just assessed your system's fitness level. You've also read very brief descriptions of the two or three strategies to level up and of the lower-level strategies that should already be in place. Perhaps this is enough for you to set in motion some wheels of change.

You'll find deeper descriptions of Levels 1-4, explanations of their strategies, and advice about executing them in Chapters 5-8. If you're short on time and want to know what specifically to do now, proceed to the chapter that corresponds to your system's current level. If you have a bit more time, and reading the text above made you suspect that the lower-level strategies aren't as "baked in" as they should be, read their chapters first.

If you're going to be closely involved in leading these changes, take the time to first read Chapters 2-4. They provide critical information about value delivery systems and explain the necessary leadership foundation for effectively executing the strategies.

For an example of a company using the strategies to move Product Development from Level 1 to 4 in only 10 months, read Appendix B. If your organization is currently on an "Agile journey" or formally undergoing an "Agile transformation," read Appendix C to learn how this book's ideas can increase its chances of success.

SUPPLEMENTARY RESOURCES: Go to DeliverBetterResultsBook.com and download "Fitness for Purpose Self-Assessment Questionnaire" and "Summary of the Five Levels."

MEET GIL BROZA

Value Delivery & Leadership Expert and Owner, 3P Vantage, Inc.
Email contact: gbroza@3PVantage.com

Gil Broza specializes in helping tech leaders deliver far better results by upgrading their Agile ways of working. He also supports their non-software colleagues in creating real business agility in their teams. He has helped over 100 large and small, private- and public-sector organizations achieve real, sustainable improvements by working with their unique contexts and focusing on mindset, culture, and leadership. Previously, Gil worked as a development manager, team leader, and programmer for many years, successfully applying Agile methods since 2001. He has served as a regular writer for the prestigious magazine *projectmanagement.com* (a PMI publication) and as a track chair for the Agile 2009, 2010, and 2016 conferences. He regularly gives keynotes and interactive talks at conferences worldwide.

Throughout his career, Gil has focused on human characteristics that prevent positive outcomes in teams and organizations. These include limiting habits, fear of change, outdated beliefs, and blind spots, among many others. In helping teams and leaders overcome these factors, he supports them in reaching ever-higher levels of performance, confidence, and accomplishment. In 2012, he published *The Human Side of Agile*, the definitive guide to leading Agile teams. In 2015, he published *The Agile Mind-Set*, helping

practitioners and leaders alike master the Agile approach and make their ways of working truly effective. And in 2019, he published *Agile for Non-Software Teams* to help managers consider, design, start, and cultivate Agile ways of working in non-software functions.

Gil provides services for establishing effective ways of working and improving value delivery. Companies also invite Gil for specialized support, such as facilitating organizational mindset workshops, delivering keynote at internal conferences, and leading sessions with executives. See his current offerings at 3PVantage.com/services.